Dikes Overtopping Kernel - Technical documentation

Generated by Doxygen 1.8.9.1

Mon Sep 5 2016 11:16:50

Contents

Chapter 1

Modules Index

1.1 Modules List

Here is a list of all modules with brief descriptions:

Chapter 2

Data Type Index

2.1 Class List

Here are the data types with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Module Documentation

4.1 dllovertopping Module Reference

Main entry for the dll DikesOvertopping.

Functions/Subroutines

- subroutine, public calculateqo (load, geometryInput, dikeHeight, modelFactors, overtopping, success, error↩ Text, verbosity, logFile)

    Subroutine that calculates the discharge needed for the Z-function DikesOvertopping Wrapper for calculateQoF↩ : convert C-like input structures to Fortran input structures.
- subroutine, public calculateqof (load, geometryF, dikeHeight, modelFactors, overtopping, success, errorText, logging)

    Subroutine that calculates the discharge needed for the Z-function DikesOvertopping.
- subroutine, public calczvalue (criticalOvertoppingRate, modelFactors, Qo, z, success, errorMessage)

    Subroutine that calculates the Z-function DikesOvertopping based on the discharge calculated with calculateQoF.
- subroutine, public validateinputc (geometryInput, dikeHeight, modelFactors, success, errorText)

    Subroutine that validates the geometry Wrapper for ValidateInputFold: convert C-like input structures to Fortran input structures.
- subroutine, public validateinputf (geometryF, dikeHeight, modelFactors, errorStruct)

    Subroutine that validates the geometry.
- subroutine, public omkeervariantf (load, geometryF, givenDischarge, dikeHeight, modelFactors, overtopping, success, errorText, logging)

    Subroutine with omkeerVariant.
- subroutine, public setlanguage (lang)

    Subroutine that sets the language for error and validation messages.
- subroutine, public getlanguage (lang)

    Subroutine that gets the language for error and validation messages.
- subroutine, public versionnumber (version)

    Subroutine that delivers the version number.
- type(overtoppinggeometrytypef) function geometry_c_f (geometryInput)

    Private subroutine that converts geometry from c-pointer to fortran struct.

4.1.1 Detailed Description

Main entry for the dll DikesOvertopping.

4.1.2 Function/Subroutine Documentation

4.1.2.1 subroutine, public dllovertopping::calculateqo (type(tpload), intent(in) *load,* type(overtoppinggeometrytype), intent(in) *geometryInput,* real(kind=wp), intent(in) *dikeHeight,* type(tpovertoppinginput), intent(inout) *modelFactors,* type (tpovertopping), intent(out) *overtopping,* logical, intent(out) *success,* character(len=∗), intent(out) *errorText,* integer, intent(in) *verbosity,* character(len=∗), intent(in) *logFile*)

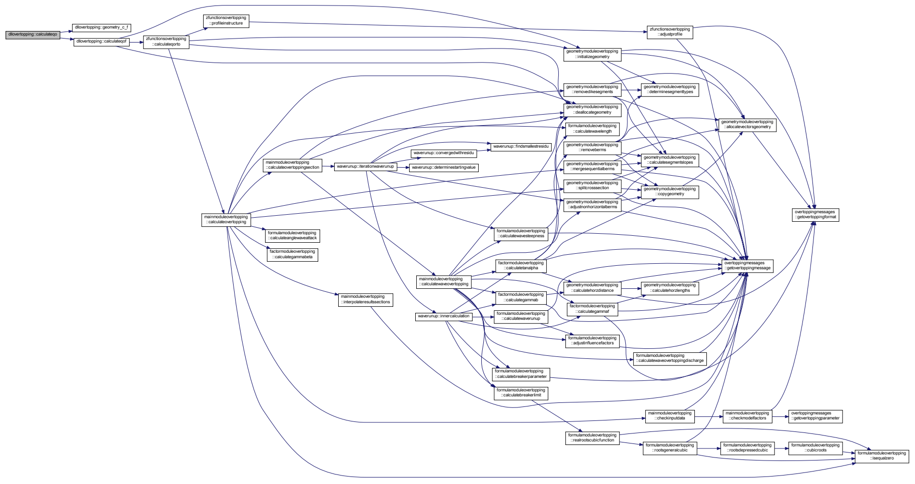
Subroutine that calculates the discharge needed for the Z-function DikesOvertopping Wrapper for calculateQoF: convert C-like input structures to Fortran input structures.

Parameters

`in`	*geometryinput*	struct with geometry and roughness as c-pointers
`in`	*load*	struct with waterlevel and wave parameters
`in`	*dikeheight*	dike height
`in,out`	*modelfactors*	struct with modelfactors
`out`	*overtopping*	structure with overtopping results
`out`	*success*	flag for success
`out`	*errortext*	error message (only set if not successful)
`in`	*verbosity*	level of verbosity
`in`	*logfile*	filename of logfile

Definition at line 65 of file dllOvertopping.f90.

Here is the call graph for this function:


dllovertopping::calculateqo
dllovertopping::geometry_c_f
dllovertopping::calculateqof
zfunctionsovertopping::profileinstructure
zfunctionsovertopping::calculateqorto
zfunctionsovertopping::adjustprofile
geometrymoduleovertopping::initializegeometry
geometrymoduleovertopping::removedikesegments
geometrymoduleovertopping::determinesegmenttypes
geometrymoduleovertopping::deallocategeometry
formulamoduleovertopping::calculatewavelength
geometrymoduleovertopping::allocatevectorsgeometry
waverunup::convergedwithresidu
waverunup::findsmallestresidu
mainmoduleovertopping::calculateovertoppingsection
waverunup::iterationwaverunup
waverunup::determinestartingvalue
geometrymoduleovertopping::removeberms
geometrymoduleovertopping::mergesequentialberms
geometrymoduleovertopping::calculatesegmentslopes
geometrymoduleovertopping::splitcrosssection
geometrymoduleovertopping::copygeometry
geometrymoduleovertopping::adjustnonhorizontalberms
overtoppingmessages::getovertoppingformat
mainmoduleovertopping::calculateovertopping
formulamoduleovertopping::calculateanglewaveattack
factormoduleovertopping::calculategammabeta
formulamoduleovertopping::calculatewavesteepness
factormoduleovertopping::calculatetanalpha
overtoppingmessages::getovertoppingmessage
mainmoduleovertopping::calculatewaveovertopping
geometrymoduleovertopping::calculatehorzdistance
geometrymoduleovertopping::calculatehorzlengths
mainmoduleovertopping::interpolateresultssections
factormoduleovertopping::calculategammab
factormoduleovertopping::calculategammaf
waverunup::innercalculation
formulamoduleovertopping::calculatewaverunup
formulamoduleovertopping::adjustinfluencefactors
formulamoduleovertopping::calculatewaveovertoppingdischarge
formulamoduleovertopping::calculatebreakerparameter
formulamoduleovertopping::calculatebreakerlimit
mainmoduleovertopping::checkinputdata
mainmoduleovertopping::checkmodelfactors
overtoppingmessages::getovertoppingparameter
formulamoduleovertopping::realrootscubicfunction
formulamoduleovertopping::rootsgeneralcubic
formulamoduleovertopping::rootsdepressedcubic
formulamoduleovertopping::cubicroots
formulamoduleovertopping::isequalzero

4.1.2.2 subroutine, public dllovertopping::calculateqof (type(tpload), intent(in) *load,* type(overtoppinggeometrytypef), intent(in) *geometryF,* real(kind=wp), intent(in) *dikeHeight,* type(tpovertoppinginput), intent(inout) *modelFactors,* type (tpovertopping), intent(out) *overtopping,* logical, intent(out) *success,* character(len=∗), intent(out) *errorText,* type(tlogging), intent(in) *logging*)

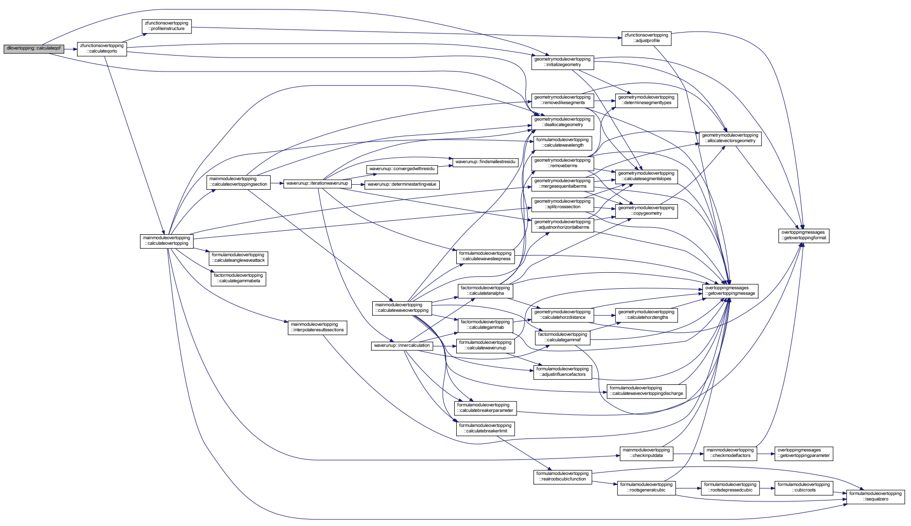
Subroutine that calculates the discharge needed for the Z-function DikesOvertopping.

Parameters

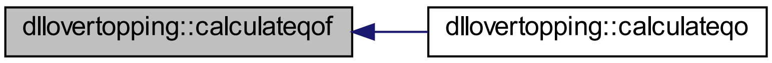
`in`	*geometryf*	struct with geometry and roughness
`in`	*load*	struct with waterlevel and wave parameters
`in`	*dikeheight*	dike height
`in,out`	*modelfactors*	struct with modelFactors
`out`	*overtopping*	structure with overtopping results
`out`	*success*	flag for success
`out`	*errortext*	error message (only set if not successful)
`in`	*logging*	logging struct

Definition at line 96 of file dllOvertopping.f90.

Here is the call graph for this function:


dllovertopping::calculateqof
zfunctionsovertopping ::profileinstructure
zfunctionsovertopping ::calculateqorto
mainmoduleovertopping ::calculateovertopping
mainmoduleovertopping ::calculateovertoppingsection
waverunup::iterationwaverunup
waverunup::convergedwithresidu
waverunup::determinestartingvalue
waverunup::findsmallestresidu
formulamoduleovertopping ::calculateanglewaveattack
factormoduleovertopping ::calculategammabeta
mainmoduleovertopping ::interpolateresultssections
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
zfunctionsovertopping ::adjustprofile
geometrymoduleovertopping ::initializegeometry
geometrymoduleovertopping ::removedikesegments
geometrymoduleovertopping ::determinesegmenttypes
geometrymoduleovertopping ::deallocategeometry
formulamoduleovertopping ::calculatewavelength
geometrymoduleovertopping ::allocatevectorsgeometry
geometrymoduleovertopping ::removeberms
geometrymoduleovertopping ::calculatesegmentslopes
geometrymoduleovertopping ::mergesequentialberms
geometrymoduleovertopping ::splitcrosssection
geometrymoduleovertopping ::copygeometry
geometrymoduleovertopping ::adjustnonhorizontalberms
overtoppingmessages ::getovertoppingformat
formulamoduleovertopping ::calculatewavesteepness
factormoduleovertopping ::calculatetanalpha
overtoppingmessages ::getovertoppingmessage
geometrymoduleovertopping ::calculatehorzdistance
geometrymoduleovertopping ::calculatehorzlengths
factormoduleovertopping ::calculategammab
factormoduleovertopping ::calculategammaf
formulamoduleovertopping ::calculatewaverunup
formulamoduleovertopping ::adjustinfluencefactors
formulamoduleovertopping ::calculatewaveovertoppingdischarge
formulamoduleovertopping ::calculatebreakerparameter
formulamoduleovertopping ::calculatebreakerlimit
mainmoduleovertopping ::checkinputdata
mainmoduleovertopping ::checkmodelfactors
overtoppingmessages ::getovertoppingparameter
formulamoduleovertopping ::realrootscubicfunction
formulamoduleovertopping ::rootsgeneralcubic
formulamoduleovertopping ::rootsdepressedcubic
formulamoduleovertopping ::cubicroots
formulamoduleovertopping ::isequalzero

Here is the caller graph for this function:



4.1.2.3 subroutine, public dllovertopping::calczvalue (real(kind=wp), intent(in) *criticalOvertoppingRate*, type(tpovertoppinginput), intent(inout) *modelFactors*, real(kind=wp), intent(in) *Qo*, real(kind=wp), intent(out) *z*, logical, intent(out) *success*, character(len=∗), intent(out) *errorMessage*)

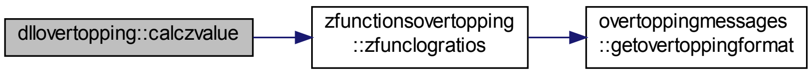
Subroutine that calculates the Z-function DikesOvertopping based on the discharge calculated with calculateQoF.

Parameters

`in`	*criticalovertop-pingrate*	critical overtoppingrate
`in,out`	*modelfactors*	struct with modelfactors
`in`	*qo*	calculated discharge
`out`	*z*	z value
`out`	*errormessage*	error message (only if not successful)
`out`	*success*	flag for success

Definition at line 129 of file dllOvertopping.f90.

Here is the call graph for this function:


dllovertopping::calczvalue
zfunctionsovertopping ::zfunclogratios
overtoppingmessages ::getovertoppingformat

4.1.2.4 type(overtoppinggeometrytypef) function dllovertopping::geometry_c_f (type(overtoppinggeometrytype), intent(in) *geometryInput*) [private]

Private subroutine that converts geometry from c-pointer to fortran struct.

Parameters

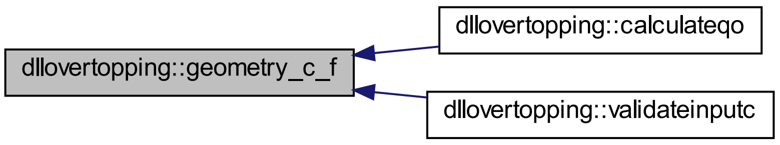
in	*geometryinput*	struct with geometry and roughness as c-pointers

Returns

fortran struct with geometry and roughness

Definition at line 350 of file dllOvertopping.f90.

Here is the caller graph for this function:


dllovertopping::geometry_c_f
dllovertopping::calculateqo
dllovertopping::validateinputc

4.1.2.5 subroutine, public dllovertopping::getlanguage (character(len=∗), intent(out) *lang*)

Subroutine that gets the language for error and validation messages.

Definition at line 316 of file dllOvertopping.f90.

4.1.2.6 subroutine, public dllovertopping::omkeervariantf (type(tpload), intent(in) *load,* type(overtoppinggeometrytypef), intent(in) *geometryF,* real(kind=wp), intent(in) *givenDischarge,* real(kind=wp), intent(out) *dikeHeight,* type(tpovertoppinginput), intent(inout) *modelFactors,* type (tpovertopping), intent(inout) *overtopping,* logical, intent(out) *success,* character(len=∗), intent(out) *errorText,* type(tlogging), intent(in) *logging*)

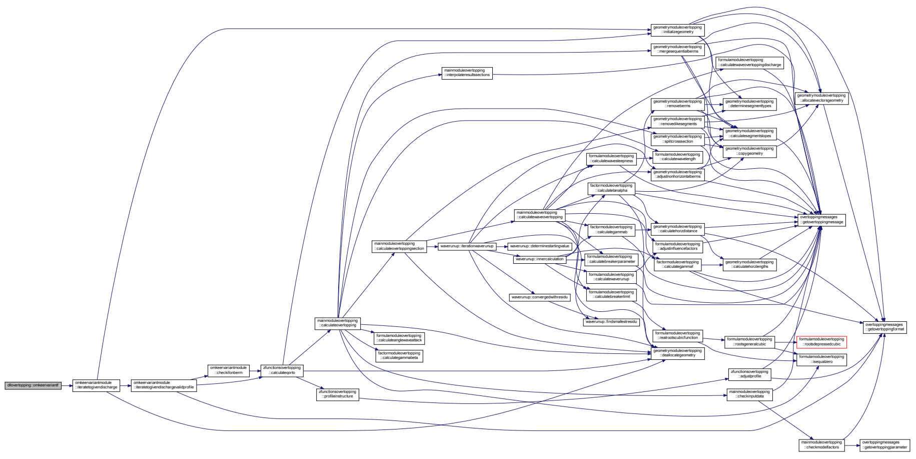
Subroutine with omkeerVariant.

Parameters

`in`	*geometryf*	struct with geometry and roughness
`in`	*load*	struct with waterlevel and wave parameters
`in`	*givendischarge*	discharge to iterate to
`out`	*dikeheight*	dike height
`in,out`	*modelfactors*	struct with modelFactors
`in,out`	*overtopping*	structure with overtopping results
`out`	*success*	flag for success
`out`	*errortext*	error message (only set if not successful)
`in`	*logging*	logging struct

Definition at line 280 of file dllOvertopping.f90.

Here is the call graph for this function:



4.1.2.7 subroutine, public dllovertopping::setlanguage (character(len=∗), intent(in) *lang*)

Subroutine that sets the language for error and validation messages.

Definition at line 303 of file dllOvertopping.f90.

4.1.2.8 subroutine, public dllovertopping::validateinputc (type(overtoppinggeometrytype), intent(in) *geometryInput,* real(kind=wp), intent(in) *dikeHeight,* type(tpovertoppinginput), intent(inout) *modelFactors,* logical, intent(out) *success,* character(len=∗), intent(out) *errorText*)

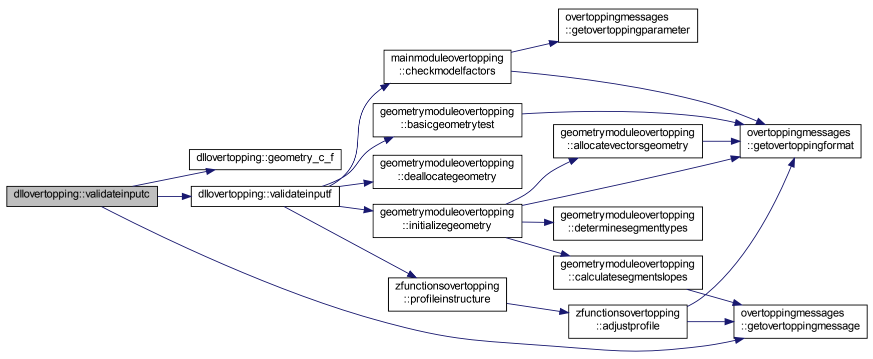
Subroutine that validates the geometry Wrapper for ValidateInputFold: convert C-like input structures to Fortran input structures.

Parameters

`in`	*geometryinput*	struct with geometry and roughness as c-pointers
`in`	*dikeheight*	dike height
`in,out`	*modelfactors*	struct with modelfactors
`out`	*success*	flag for success
`out`	*errortext*	error message (only set if not successful)

Definition at line 149 of file dllOvertopping.f90.

Here is the call graph for this function:


dllovertopping::validateinputc
dllovertopping::geometry_c_f
dllovertopping::validateinputf
mainmoduleovertopping ::checkmodelfactors
geometrymoduleovertopping ::basicgeometrytest
geometrymoduleovertopping ::deallocategeometry
geometrymoduleovertopping ::initializegeometry
zfunctionsovertopping ::profileinstructure
overtoppingmessages ::getovertoppingparameter
geometrymoduleovertopping ::allocatevectorsgeometry
geometrymoduleovertopping ::determinesegmenttypes
geometrymoduleovertopping ::calculatesegmentslopes
zfunctionsovertopping ::adjustprofile
overtoppingmessages ::getovertoppingformat
overtoppingmessages ::getovertoppingmessage

4.1.2.9 subroutine, public dllovertopping::validateinputf (type(overtoppinggeometrytypef), intent(in) *geometryF,* real(kind=wp), intent(in) *dikeHeight,* type(tpovertoppinginput), intent(inout) *modelFactors,* type(terrormessages), intent(inout) *errorStruct*)

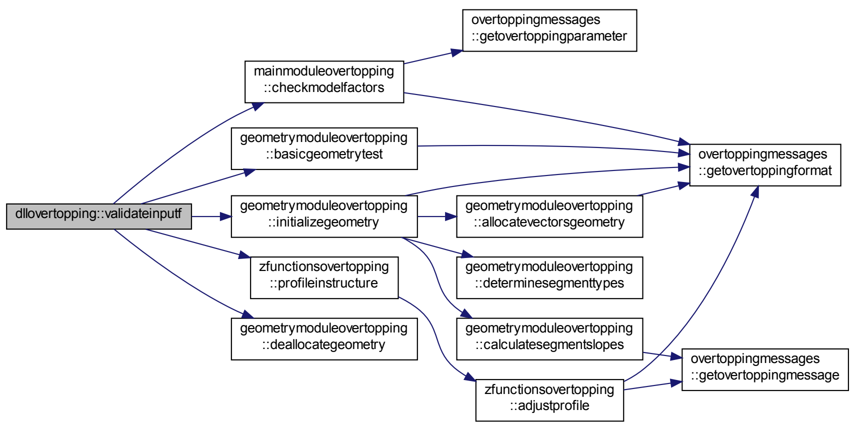
Subroutine that validates the geometry.

Parameters

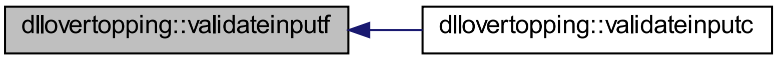
`in`	*geometryf*	struct with geometry and roughness
`in`	*dikeheight*	dike height
`in,out`	*modelfactors*	struct with modelFactors
`in,out`	*errorstruct*	error message (only set if not successful)

Definition at line 200 of file dllOvertopping.f90.

Here is the call graph for this function:


dllovertopping::validateinputf
mainmoduleovertopping ::checkmodelfactors
geometrymoduleovertopping ::basicgeometrytest
geometrymoduleovertopping ::initializegeometry
zfunctionsovertopping ::profileinstructure
geometrymoduleovertopping ::deallocategeometry
overtoppingmessages ::getovertoppingparameter
geometrymoduleovertopping ::allocatevectorsgeometry
geometrymoduleovertopping ::determinesegmenttypes
geometrymoduleovertopping ::calculatesegmentslopes
zfunctionsovertopping ::adjustprofile
overtoppingmessages ::getovertoppingformat
overtoppingmessages ::getovertoppingmessage

Here is the caller graph for this function:


dllovertopping::validateinputf
dllovertopping::validateinputc

4.1.2.10 subroutine, public dllovertopping::versionnumber (character(len=∗), intent(out) *version*)

Subroutine that delivers the version number.

Parameters

`out`	*version*	version number

Definition at line 328 of file dllOvertopping.f90.

4.2 factormoduleovertopping Module Reference

functions for the slope angle and influence factors

Functions/Subroutines

- subroutine, public calculatetanalpha (h, Hm0, z2, geometry, tanAlpha, succes, errorMessage)

  calculateTanAlpha representative slope angle
- subroutine, public calculategammabeta (Hm0, Tm_10, beta, gammaBeta_z, gammaBeta_o)

  calculateGammaBeta influence factor angle of wave attack
- subroutine, public calculategammaf (h, ksi0, ksi0Limit, gammaB, z2, geometry, gammaF, succes, error↩ Message)

  calculateGammaF influence factor roughness
- subroutine, public calculategammab (h, Hm0, z2, geometry, gammaB, succes, errorMessage)

  calculateGammaB influence factor berms

4.2.1 Detailed Description

functions for the slope angle and influence factors

4.2.2 Function/Subroutine Documentation

4.2.2.1 subroutine, public factormoduleovertopping::calculategammab (real(kind=wp), intent(in) *h*, real(kind=wp), intent(in) *Hm0*, real(kind=wp), intent(in) *z2*, type(tpgeometry), intent(in) *geometry*, real(kind=wp), intent(out) *gammaB*, logical, intent(out) *succes*, character(len=∗), intent(out) *errorMessage*)

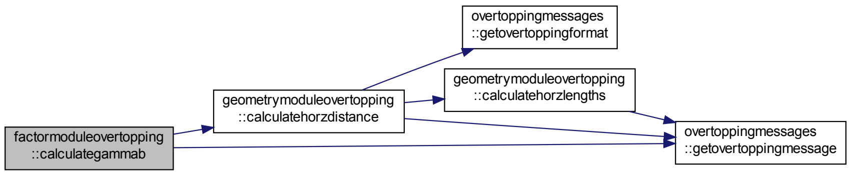
calculateGammaB influence factor berms

Parameters

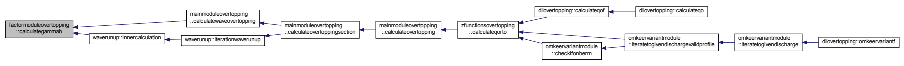
`in`	*h*	local water level (m+NAP)
`in`	*hm0*	significant wave height (m)
`in`	*z2*	2% wave run-up (m)
`in`	*geometry*	structure with geometry data
`out`	*gammab*	influence factor berms
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 310 of file factorModuleOvertopping.f90.

Here is the call graph for this function:


factormoduleovertopping ::calculategammab
geometrymoduleovertopping ::calculatehorzdistance
overtoppingmessages ::getovertoppingformat
geometrymoduleovertopping ::calculatehorzlengths
overtoppingmessages ::getovertoppingmessage

Here is the caller graph for this function:


factormoduleovertopping ::calculategammab
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.2.2.2 subroutine, public factormoduleovertopping::calculategammabeta (real(kind=wp), intent(inout) *Hm0*, real(kind=wp), intent(inout) *Tm_10*, real(kind=wp), intent(in) *beta*, real(kind=wp), intent(out) *gammaBeta_z*, real(kind=wp), intent(out) *gammaBeta_o*)

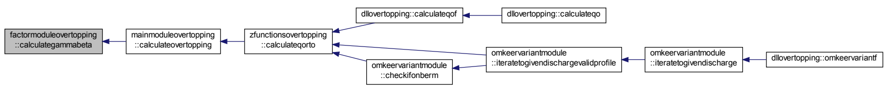
calculateGammaBeta influence factor angle of wave attack

Parameters

`in,out`	*hm0*	significant wave height (m)
`in,out`	*tm_10*	spectral wave period (s)
`in`	*beta*	angle of wave attack (degree)
`out`	*gammabeta_z*	influence factor angle of wave attack 2% wave run-up
`out`	*gammabeta_o*	influence factor angle of wave attack overtopping

Definition at line 139 of file factorModuleOvertopping.f90.

Here is the caller graph for this function:


factormoduleovertopping ::calculategammabeta
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.2.2.3 subroutine, public factormoduleovertopping::calculategammaf (real(kind=wp), intent(in) *h,* real(kind=wp), intent(in) *ksi0,* real(kind=wp), intent(in) *ksi0Limit,* real(kind=wp), intent(in) *gammaB,* real(kind=wp), intent(in) *z2,* type(tpgeometry), intent(in) *geometry,* real(kind=wp), intent(out) *gammaF,* logical, intent(out) *succes,* character(len=*), intent(out) *errorMessage*)

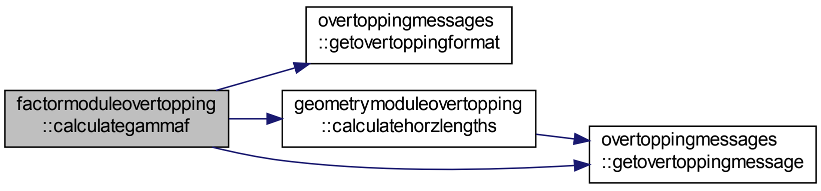
calculateGammaF influence factor roughness

Parameters

`in`	*h*	local water level (m+NAP)
`in`	*ksi0*	breaker parameter
`in`	*ksi0limit*	limit value breaker parameter
`in`	*gammab*	influence factor berms
`in`	*z2*	2% wave run-up (m)
`in`	*geometry*	structure with geometry data
`out`	*gammaf*	influence factor roughness
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

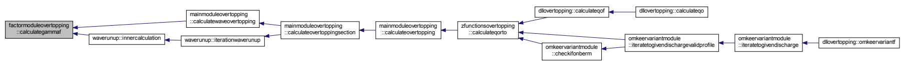
Definition at line 179 of file factorModuleOvertopping.f90.

Here is the call graph for this function:



overtoppingmessages ::getovertoppingformat
factormoduleovertopping ::calculategammaf
geometrymoduleovertopping ::calculatehorzlengths
overtoppingmessages ::getovertoppingmessage

Here is the caller graph for this function:



factormoduleovertopping ::calculategammaf
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.2.2.4 subroutine, public factormoduleovertopping::calculatetanalpha (real(kind=wp), intent(in) *h*, real(kind=wp), intent(in) *Hm0*, real(kind=wp), intent(in) *z2*, type(tpgeometry), intent(in) *geometry*, real(kind=wp), intent(out) *tanAlpha*, logical, intent(out) *succes*, character(len=∗), intent(out) *errorMessage*)

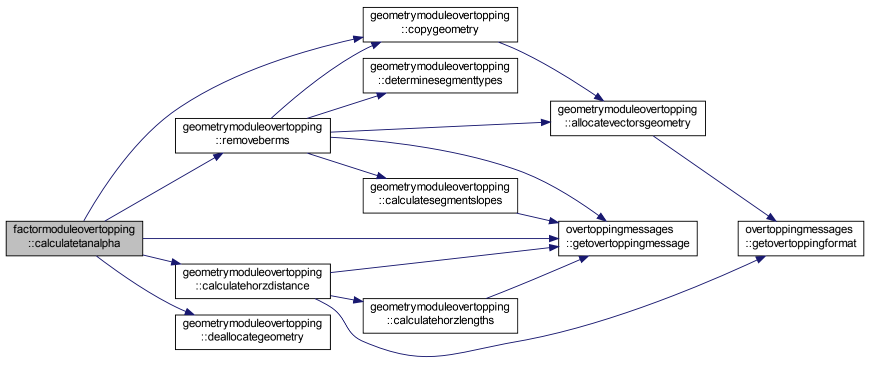
calculateTanAlpha representative slope angle

Parameters

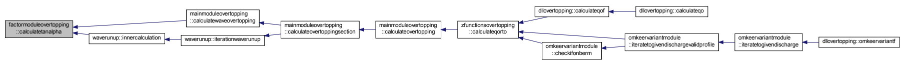
`in`	*h*	local water level (m+NAP)
`in`	*hm0*	significant wave height (m)
`in`	*z2*	2% wave run-up (m)
`in`	*geometry*	structure with geometry data
`out`	*tanalpha*	representative slope angle
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 60 of file factorModuleOvertopping.f90.

Here is the call graph for this function:


geometrymoduleovertopping ::copygeometry
geometrymoduleovertopping ::determinesegmenttypes
geometrymoduleovertopping ::allocatevectorsgeometry
geometrymoduleovertopping ::removeberms
geometrymoduleovertopping ::calculatesegmentslopes
factormoduleovertopping ::calculatetanalpha
overtoppingmessages ::getovertoppingmessage
overtoppingmessages ::getovertoppingformat
geometrymoduleovertopping ::calculatehorzdistance
geometrymoduleovertopping ::calculatehorzlengths
geometrymoduleovertopping ::deallocategeometry

Here is the caller graph for this function:


factormoduleovertopping ::calculatetanalpha
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3 formulamoduleovertopping Module Reference

the core computations for Dikes Overtopping

Functions/Subroutines

- subroutine, public calculatewaverunup (Hm0, ksi0, ksi0Limit, gammaB, gammaF, gammaBeta, modelFactors, z2, succes, errorMessage)

  calculateWaveRunup: calculate wave runup
- subroutine, public calculatewaveovertoppingdischarge (h, Hm0, tanAlpha, gammaB, gammaF, gammaBeta, ksi0, hCrest, modelFactors, Qo, succes, errorMessage)

  calculateWaveOvertoppingDischarge: calculate the wave overtopping discharge
- subroutine, public calculatewavelength (Tm_10, L0)

  calculateWaveLength: calculate the wave length
- subroutine, public calculatewavesteepness (Hm0, Tm_10, s0, succes, errorMessage)

  calculateWaveSteepness: calculate the wave steepness
- subroutine, public calculatebreakerparameter (tanAlpha, s0, ksi0, succes, errorMessage)

  calculateBreakerParameter: calculate the breaker parameter
- subroutine, public calculateanglewaveattack (phi, psi, beta)

  calculateAngleWaveAttack: calculate the angle of wave attack
- subroutine, public calculatebreakerlimit (gammaB, ksi0Limit, succes, errorMessage)

  calculateBreakerLimit: calculate the breaker limit
- subroutine, public adjustinfluencefactors (gammaB, gammaF, gammaBeta, gammaBetaType, ksi0, ksi0Limit, succes, errorMessage)

  adjustInfluenceFactors: adjust the influence factors
- subroutine realrootscubicfunction (a, b, c, d, N, x, succes, errorMessage)

  realRootsCubicFunction: calculate the roots of a cubic function
- subroutine rootsgeneralcubic (a, b, c, d, z, succes, errorMessage)

rootsGeneralCubic: calculate the roots of a generic cubic function

- subroutine rootsdepressedcubic (p, q, z)

  rootsDepressedCubic: calculate the roots of a depressed cubic function

- subroutine cubicroots (z, roots)

  cubicRoots: calculate the roots of a cubic function

- logical function, public isequalreal (x1, x2)

  isEqualReal: are two reals (almost) equal

- logical function, public isequalzero (x)

  isEqualZero: is a real (almost) zero

4.3.1 Detailed Description

the core computations for Dikes Overtopping

4.3.2 Function/Subroutine Documentation

4.3.2.1 subroutine, public formulamoduleovertopping::adjustinfluencefactors (real(kind=wp), intent(inout) *gammaB,* real(kind=wp), intent(inout) *gammaF,* real(kind=wp), intent(inout) *gammaBeta,* integer, intent(in) *gammaBetaType,* real(kind=wp), intent(in) *ksi0,* real(kind=wp), intent(in) *ksi0Limit,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

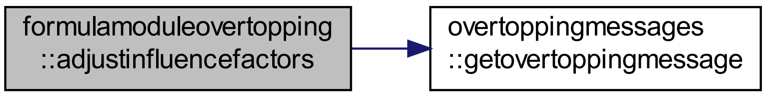
adjustInfluenceFactors: adjust the influence factors

Parameters

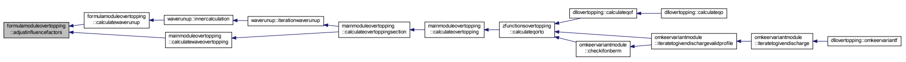
`in,out`	*gammab*	influence factor berms
`in,out`	*gammaf*	influence factor roughness
`in,out`	*gammabeta*	influence factor angle of wave attack
`in`	*gammabetatype*	type influence factor angle of wave attack: 1 = wave run-up, 2 = overtopping
`in`	*ksi0*	breaker parameter
`in`	*ksi0limit*	limit value breaker parameter
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 403 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:


formulamoduleovertopping ::adjustinfluencefactors
overtoppingmessages ::getovertoppingmessage

Here is the caller graph for this function:


formulamoduleovertopping ::adjustinfluencefactors
formulamoduleovertopping ::calculatewaverunup
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculatewaveovertopping
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3.2.2 subroutine, public formulamoduleovertopping::calculateanglewaveattack (real(kind=wp), intent(in) *phi,* real(kind=wp), intent(in) *psi,* real(kind=wp), intent(out) *beta*)

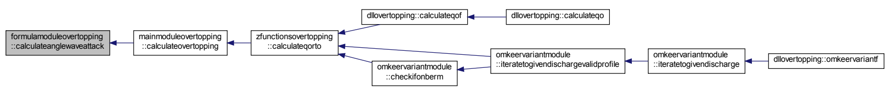
calculateAngleWaveAttack: calculate the angle of wave attack

Parameters

in	*phi*	wave direction (degree)
in	*psi*	dike normal (degree)
out	*beta*	angle of wave attack (degree)

Definition at line 309 of file formulaModuleOvertopping.f90.

Here is the caller graph for this function:

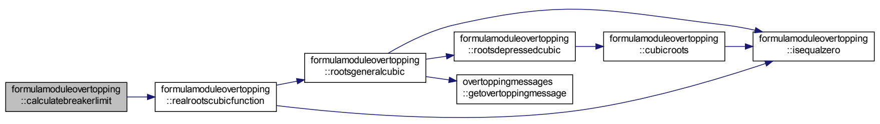

formulamoduleovertopping ::calculateanglewaveattack
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3.2.3 subroutine, public formulamoduleovertopping::calculatebreakerlimit (real(kind=wp), intent(in) *gammaB*, real(kind=wp), intent(out) *ksi0Limit*, logical, intent(out) *succes*, character(len=∗), intent(out) *errorMessage*)

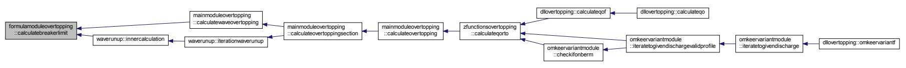
calculateBreakerLimit: calculate the breaker limit

Definition at line 332 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:


formulamoduleovertopping ::calculatebreakerlimit
formulamoduleovertopping ::realrootscubicfunction
formulamoduleovertopping ::rootsgeneralcubic
formulamoduleovertopping ::rootsdepressedcubic
overtoppingmessages ::getovertoppingmessage
formulamoduleovertopping ::cubicroots
formulamoduleovertopping ::isequalzero

Here is the caller graph for this function:


formulamoduleovertopping ::calculatebreakerlimit
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3.2.4 subroutine, public formulamoduleovertopping::calculatebreakerparameter (real(kind=wp), intent(in) *tanAlpha*, real(kind=wp), intent(in) *s0*, real(kind=wp), intent(out) *ksi0*, logical, intent(out) *succes*, character(len=∗), intent(out) *errorMessage*)

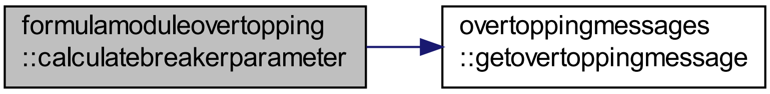
calculateBreakerParameter: calculate the breaker parameter

Parameters

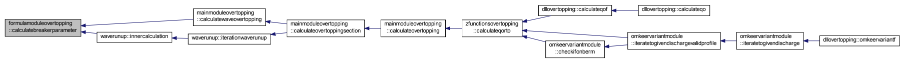
in	*tanalpha*	representative slope angle
in	*s0*	wave steepness
out	*ksi0*	breaker parameter
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 268 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:


formulamoduleovertopping ::calculatebreakerparameter
overtoppingmessages ::getovertoppingmessage

Here is the caller graph for this function:


formulamoduleovertopping ::calculatebreakerparameter
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3.2.5 subroutine, public formulamoduleovertopping::calculatewavelength (real(kind=wp), intent(in) *Tm_10*, real(kind=wp), intent(out) *L0*)

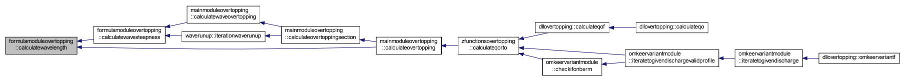
calculateWaveLength: calculate the wave length

Parameters

`in`	*tm_10*	spectral wave period (s)
`out`	*l0*	wave length (m)

Definition at line 205 of file formulaModuleOvertopping.f90.

Here is the caller graph for this function:


formulamoduleovertopping ::calculatewavelength
formulamoduleovertopping ::calculatewavesteepness
mainmoduleovertopping ::calculatewaveovertopping
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3.2.6 subroutine, public formulamoduleovertopping::calculatewaveovertoppingdischarge (real(kind=wp), intent(in) *h*, real(kind=wp), intent(in) *Hm0*, real(kind=wp), intent(in) *tanAlpha*, real(kind=wp), intent(in) *gammaB*, real(kind=wp), intent(in) *gammaF*, real(kind=wp), intent(in) *gammaBeta*, real(kind=wp), intent(in) *ksi0*, real(kind=wp), intent(in) *hCrest*, type(tpovertoppinginput), intent(in) *modelFactors*, real(kind=wp), intent(out) *Qo*, logical, intent(out) *succes*, character(len=∗), intent(out) *errorMessage*)

calculateWaveOvertoppingDischarge: calculate the wave overtopping discharge

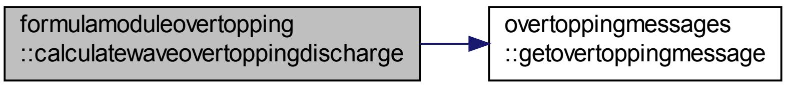
Parameters

`in`	*h*	local water level (m+NAP)
`in`	*hm0*	significant wave height (m)
`in`	*tanalpha*	representative slope angle
`in`	*gammab*	influence factor berms

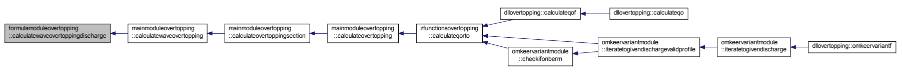
in	*gammaf*	influence factor roughness
in	*gammabeta*	influence factor angle of wave attack
in	*ksi0*	breaker parameter
in	*hcrest*	crest level (m+NAP)
in	*modelfactors*	structure with model factors
out	*qo*	wave overtopping discharge (l/m per s)
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 108 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:


formulamoduleovertopping ::calculatewaveovertoppingdischarge
overtoppingmessages ::getovertoppingmessage

Here is the caller graph for this function:


formulamoduleovertopping ::calculatewaveovertoppingdischarge
mainmoduleovertopping ::calculatewaveovertopping
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3.2.7 subroutine, public formulamoduleovertopping::calculatewaverunup (real(kind=wp), intent(in) *Hm0,* real(kind=wp), intent(in) *ksi0,* real(kind=wp), intent(in) *ksi0Limit,* real(kind=wp), intent(inout) *gammaB,* real(kind=wp), intent(inout) *gammaF,* real(kind=wp), intent(inout) *gammaBeta,* type (tpovertoppinginput), intent(in) *modelFactors,* real(kind=wp), intent(out) *z2,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

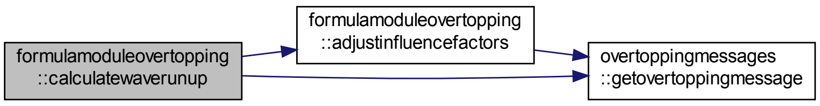
calculateWaveRunup: calculate wave runup

Parameters

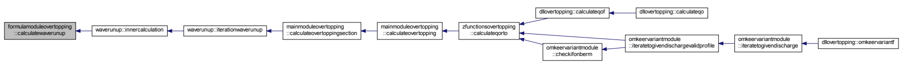
in	*hm0*	significant wave height (m)
in	*ksi0*	breaker parameter
in	*ksi0limit*	limit value breaker parameter
in,out	*gammab*	influence factor berms
in,out	*gammaf*	influence factor roughness
in,out	*gammabeta*	influence factor angle of wave attack
in	*modelfactors*	structure with model factors
out	*z2*	2% wave run-up (m)
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 59 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:


formulamoduleovertopping ::calculatewaverunup
formulamoduleovertopping ::adjustinfluencefactors
overtoppingmessages ::getovertoppingmessage

Here is the caller graph for this function:


formulamoduleovertopping ::calculatewaverunup
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3.2.8 subroutine, public formulamoduleovertopping::calculatewavesteepness (real(kind=wp), intent(in) *Hm0*, real(kind=wp), intent(in) *Tm_10*, real(kind=wp), intent(out) *s0*, logical, intent(out) *succes*, character(len=∗), intent(out) *errorMessage*)

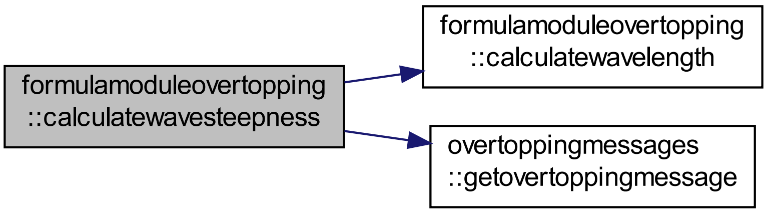
calculateWaveSteepness: calculate the wave steepness

Parameters

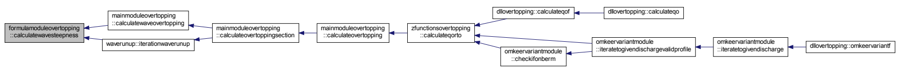
`in`	*hm0*	significant wave height (m)
`in`	*tm_10*	spectral wave period (s)
`out`	*s0*	wave steepness
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 226 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:


formulamoduleovertopping ::calculatewavesteepness
formulamoduleovertopping ::calculatewavelength
overtoppingmessages ::getovertoppingmessage

Here is the caller graph for this function:


formulamoduleovertopping ::calculatewavesteepness
mainmoduleovertopping ::calculatewaveovertopping
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3.2.9 subroutine formulamoduleovertopping::cubicroots (double complex, intent(in) *z*, double complex, dimension(3), intent(out) ***roots***) `[private]`

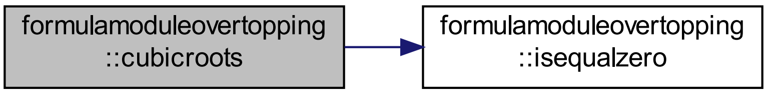
cubicRoots: calculate the roots of a cubic function

Parameters

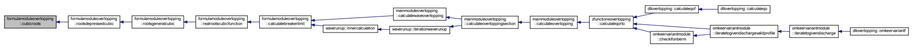
in	*z*	complex number
out	*roots*	cubic roots

Definition at line 651 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:


formulamoduleovertopping ::cubicroots
formulamoduleovertopping ::isequalzero

Here is the caller graph for this function:


formulamoduleovertopping ::cubicroots
formulamoduleovertopping ::rootsdepressedcubic
formulamoduleovertopping ::rootsgeneralcubic
formulamoduleovertopping ::realrootscubicfunction
formulamoduleovertopping ::calculatebreakerlimit
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3.2.10 logical function, public formulamoduleovertopping::isequalreal (real(kind=wp), intent(in) *x1*, real(kind=wp), intent(in) *x2*)

isEqualReal: are two reals (almost) equal

Parameters

in	*x1*	first real
in	*x2*	second real

Definition at line 692 of file formulaModuleOvertopping.f90.

4.3.2.11 logical function, public formulamoduleovertopping::isequalzero (real(kind=wp), intent(in) *x*)

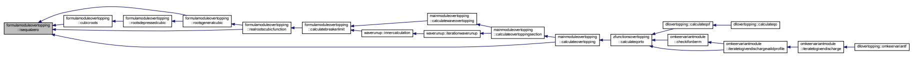
isEqualZero: is a real (almost) zero

Parameters

in	*x*	real number

Definition at line 716 of file formulaModuleOvertopping.f90.

Here is the caller graph for this function:


formulamoduleovertopping ::isequalzero
formulamoduleovertopping ::cubicroots
formulamoduleovertopping ::rootsdepressedcubic
formulamoduleovertopping ::rootsgeneralcubic
formulamoduleovertopping ::realrootscubicfunction
formulamoduleovertopping ::calculatebreakerlimit
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3.2.12 subroutine formulamoduleovertopping::realrootscubicfunction (real(kind=wp), intent(in) *a*, real(kind=wp), intent(in) *b*, real(kind=wp), intent(in) *c*, real(kind=wp), intent(in) *d*, integer, intent(out) *N*, real(kind=wp), dimension(3), intent(out) *x*, logical, intent(out) *succes*, character(len=∗), intent(out) *errorMessage*) `[private]`

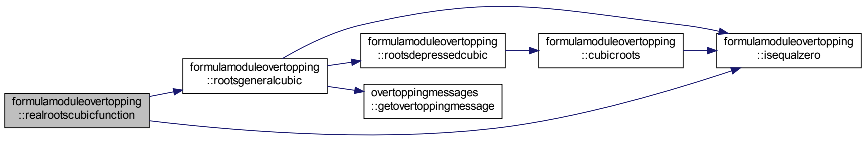
realRootsCubicFunction: calculate the roots of a cubic function

Parameters

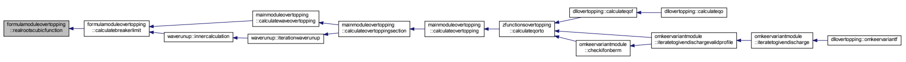
in	*a*	coefficient a cubic function
in	*b*	coefficient b cubic function
in	*c*	coefficient c cubic function
in	*d*	coefficient d cubic function
out	*n*	number of real roots cubic function
out	*x*	real roots cubic function
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 501 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:


formulamoduleovertopping ::realrootscubicfunction
formulamoduleovertopping ::rootsgeneralcubic
formulamoduleovertopping ::rootsdepressedcubic
overtoppingmessages ::getovertoppingmessage
formulamoduleovertopping ::cubicroots
formulamoduleovertopping ::isequalzero

Here is the caller graph for this function:


formulamoduleovertopping ::realrootscubicfunction
formulamoduleovertopping ::calculatebreakerlimit
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3.2.13 subroutine formulamoduleovertopping::rootsdepressedcubic (real(kind=wp), intent(in) *p*, real(kind=wp), intent(in) *q*, double complex, dimension(3), intent(out) *z*) `[private]`

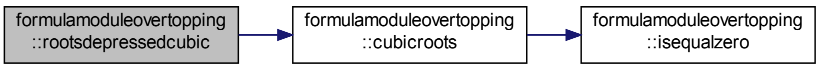
rootsDepressedCubic: calculate the roots of a depressed cubic function

Parameters

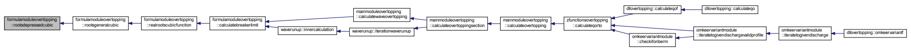
in	*p*	coefficient p depressed cubic
in	*q*	coefficient q depressed cubic
out	*z*	roots depressed cubic

Definition at line 610 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:


formulamoduleovertopping ::rootsdepressedcubic
formulamoduleovertopping ::cubicroots
formulamoduleovertopping ::isequalzero

Here is the caller graph for this function:


formulamoduleovertopping ::rootsdepressedcubic
formulamoduleovertopping ::rootsgeneralcubic
formulamoduleovertopping ::realrootscubicfunction
formulamoduleovertopping ::calculatebreakerlimit
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.3.2.14 subroutine formulamoduleovertopping::rootsgeneralcubic (real(kind=wp), intent(in) *a*, real(kind=wp), intent(in) *b*, real(kind=wp), intent(in) *c*, real(kind=wp), intent(in) *d*, double complex, dimension(3), intent(out) *z*, logical, intent(out) *succes*, character(len=∗), intent(out) *errorMessage*) `[private]`

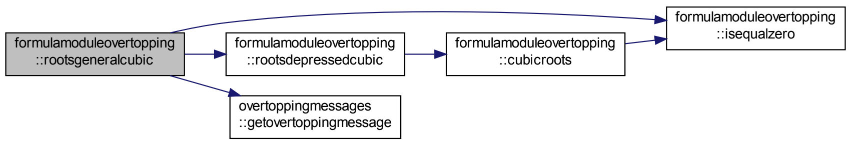
rootsGeneralCubic: calculate the roots of a generic cubic function

Parameters

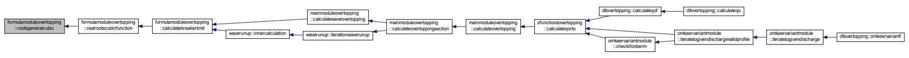
in	*a*	coefficients a cubic function
in	*b*	coefficients b cubic function
in	*c*	coefficients c cubic function
in	*d*	coefficients d cubic function
out	*z*	roots cubic function
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 557 of file formulaModuleOvertopping.f90.

Here is the call graph for this function:


formulamoduleovertopping ::rootsgeneralcubic
formulamoduleovertopping ::rootsdepressedcubic
formulamoduleovertopping ::cubicroots
formulamoduleovertopping ::isequalzero
overtoppingmessages ::getovertoppingmessage

Here is the caller graph for this function:


formulamoduleovertopping ::rootsgeneralcubic
formulamoduleovertopping ::realrootscubicfunction
formulamoduleovertopping ::calculatebreakerlimit
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.4 geometrymoduleovertopping Module Reference

core computations related to the geometry

Functions/Subroutines

- subroutine, public checkcrosssection (psi, nCoordinates, xCoordinates, yCoordinates, roughnessFactors, succes, errorMessage)

  checkCrossSection: check cross section
- subroutine, public initializegeometry (psi, nCoordinates, xCoordinates, yCoordinates, roughnessFactors, geometry, succes, errorMessage)

  initializeGeometry: initialize the geometry
- subroutine, public allocatevectorsgeometry (nCoordinates, geometry, succes, errorMessage)

  allocateVectorsGeometry: allocate the geometry vectors
- subroutine, public deallocategeometry (geometry)

  deallocateGeometry: deallocate the geometry vectors
- subroutine, public calculatesegmentslopes (geometry, succes, errorMessage)

  calculateSegmentSlopes: calculate the segment slopes
- subroutine, public determinesegmenttypes (geometry)

  determineSegmentTypes: determine the segment types
- subroutine, public copygeometry (geometry, geometryCopy, succes, errorMessage)

  copyGeometry: copy a geometry structure
- subroutine, public mergesequentialberms (geometry, geometryMergedBerms, succes, errorMessage)

  mergeSequentialBerms: merge sequential berms

- subroutine, public adjustnonhorizontalberms (geometry, geometryFlatBerms, succes, errorMessage)

  adjustNonHorizontalBerms: adjust non-horizontal berms
- subroutine, public removeberms (geometry, geometryNoBerms, succes, errorMessage)

  removeBerms: remove berms
- subroutine, public removedikesegments (geometry, index, geometryAdjusted, succes, errorMessage)

  removeDikeSegments: remove dike segments
- subroutine, public splitcrosssection (geometry, L0, NwideBerms, geometrysectionB, geometrysectionF, succes, errorMessage)

  splitCrossSection: split a cross section
- subroutine, public calculatehorzlengths (geometry, yLower, yUpper, horzLengths, succes, errorMessage)

  calculateHorzLengths: calculate horizontal lengths
- subroutine, public calculatehorzdistance (geometry, yLower, yUpper, dx, succes, errorMessage)

  calculateHorzDistance: calculate horizontal distance
- subroutine, public basicgeometrytest (geometryF, success, errorStruct)

  basicGeometryTest: test the input geometry (the adjusted geometry is checked elsewhere)

4.4.1 Detailed Description

core computations related to the geometry

4.4.2 Function/Subroutine Documentation

4.4.2.1 subroutine, public geometrymoduleovertopping::adjustnonhorizontalberms (type (tpgeometry), intent(in) *geometry,* type (tpgeometry), intent(out) *geometryFlatBerms,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

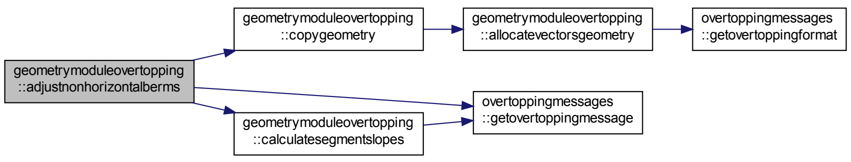
adjustNonHorizontalBerms: adjust non-horizontal berms

Parameters

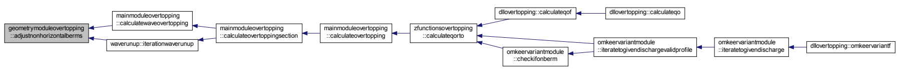
`in`	*geometry*	structure with geometry data
`out`	*geometryflat-berms*	geometry data with horizontal berms
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 550 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:


geometrymoduleovertopping ::adjustnonhorizontalberms
geometrymoduleovertopping ::copygeometry
geometrymoduleovertopping ::allocatevectorsgeometry
overtoppingmessages ::getovertoppingformat
overtoppingmessages ::getovertoppingmessage
geometrymoduleovertopping ::calculatesegmentslopes

Here is the caller graph for this function:


geometrymoduleovertopping ::adjustnonhorizontalberms
mainmoduleovertopping ::calculatewaveovertopping
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.4.2.2 subroutine, public geometrymoduleovertopping::allocatevectorsgeometry (integer, intent(in) *nCoordinates,* type (tpgeometry), intent(inout) *geometry,* logical, intent(out) *succes,* character(len=∗), intent(inout) *errorMessage*)

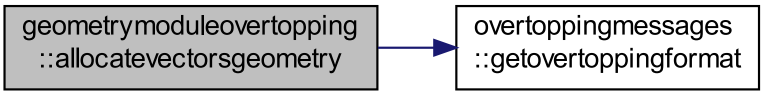
allocateVectorsGeometry: allocate the geometry vectors

Parameters

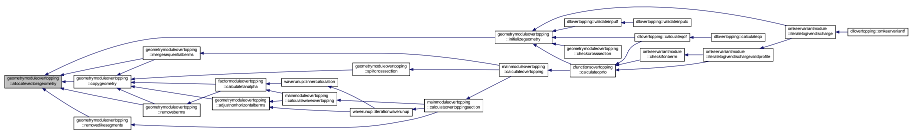
in	*ncoordinates*	number of coordinates
in,out	*geometry*	structure with geometry data
out	*succes*	succes flag
in,out	*errormessage*	error message (only set in case of error)

Definition at line 246 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:


geometrymoduleovertopping
::allocatevectorsgeometry
geometrymoduleovertopping
::mergesequentialberms
geometrymoduleovertopping
::copygeometry
geometrymoduleovertopping
::removeberms
geometrymoduleovertopping
::removedikesegments
factormoduleovertopping
::calculatetanalpha
geometrymoduleovertopping
::adjustnonhorizontalberms
waverunup::innercalculation
mainmoduleovertopping
::calculatewaveovertopping
waverunup::iterationwaverunup
geometrymoduleovertopping
::splitcrosssection
mainmoduleovertopping
::calculateovertoppingsection
geometrymoduleovertopping
::initializegeometry
mainmoduleovertopping
::calculateovertopping
zfunctionsovertopping
::calculateqorto
dllovertopping::validateinputf
dllovertopping::validateinputc
dllovertopping::calculateqof
dllovertopping::calculateqo
geometrymoduleovertopping
::checkcrosssection
omkeervariantmodule
::checkifonberm
omkeervariantmodule
::iteratetogivendischargevalidprofile
omkeervariantmodule
::iteratetogivendischarge
dllovertopping::omkeervariantf

4.4.2.3 subroutine, public geometrymoduleovertopping::basicgeometrytest (type(overtoppinggeometrytypef), intent(in) *geometryF,* logical, intent(out) *success,* type(terrormessages), intent(inout) *errorStruct*)

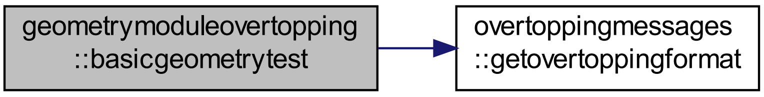
basicGeometryTest: test the input geometry (the adjusted geometry is checked elsewhere)

Parameters

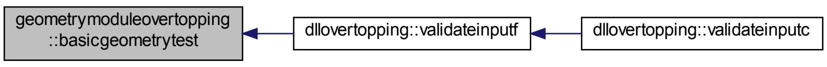
in	*geometryf*	struct with geometry and roughness
in,out	*errorstruct*	error message (only set if not successful)
out	*success*	success flag

Definition at line 1053 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:


geometrymoduleovertopping ::basicgeometrytest
dllovertopping::validateinputf
dllovertopping::validateinputc

4.4.2.4 subroutine, public geometrymoduleovertopping::calculatehorzdistance (type (tpgeometry), intent(in) *geometry,* real(kind=wp), intent(in) *yLower,* real(kind=wp), intent(in) *yUpper,* real(kind=wp), intent(out) *dx,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

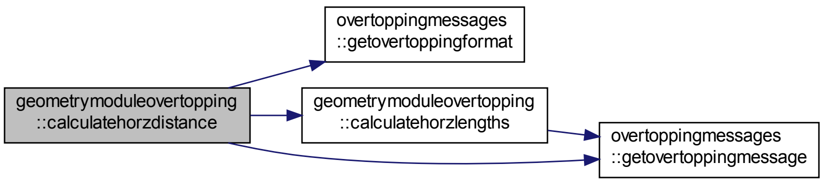
calculateHorzDistance: calculate horizontal distance

Parameters

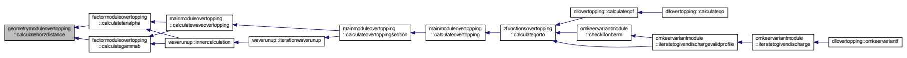
in	*geometry*	structure with geometry data
in	*ylower*	y-coordinate lower bound (m+NAP)
in	*yupper*	y-coordinate upper bound (m+NAP)
out	*dx*	horizontal distance between bounds (m)
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 1004 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:


overtoppingmessages ::getovertoppingformat
geometrymoduleovertopping ::calculatehorzdistance
geometrymoduleovertopping ::calculatehorzlengths
overtoppingmessages ::getovertoppingmessage

Here is the caller graph for this function:


geometrymoduleovertopping ::calculatehorzdistance
factormoduleovertopping ::calculatetanalpha
factormoduleovertopping ::calculategammab
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.4.2.5 subroutine, public geometrymoduleovertopping::calculatehorzlengths (type (tpgeometry), intent(in) *geometry,* real(kind=wp), intent(in) *yLower,* real(kind=wp), intent(in) *yUpper,* real(kind=wp), dimension(geometry%ncoordinates-1), intent(out) *horzLengths,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

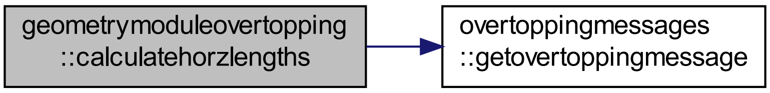
calculateHorzLengths: calculate horizontal lengths

Parameters

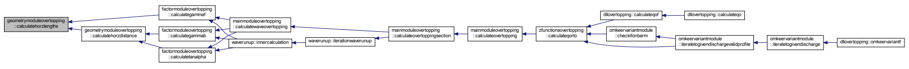
`in`	*geometry*	structure with geometry data
`in`	*ylower*	y-coord. lower bound (m+NAP)
`in`	*yupper*	y-coord. upper bound (m+NAP)
`out`	*horzlengths*	horizontal lengths segments (m)
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 908 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.4.2.6 subroutine, public geometrymoduleovertopping::calculatesegmentslopes (type (tpgeometry), intent(inout) *geometry*, logical, intent(out) *succes*, character(len=∗), intent(out) *errorMessage*)

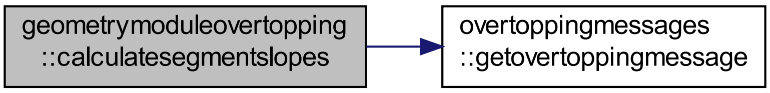
calculateSegmentSlopes: calculate the segment slopes

Parameters

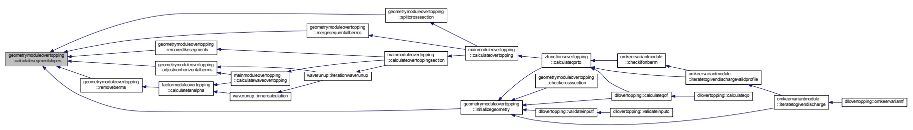
`in,out`	*geometry*	structure with geometry data
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 308 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:


geometrymoduleovertopping ::calculatesegmentslopes
geometrymoduleovertopping ::splitcrosssection
geometrymoduleovertopping ::mergesequentialberms
geometrymoduleovertopping ::removedikesegments
geometrymoduleovertopping ::adjustnonhorizontalberms
geometrymoduleovertopping ::removeberms
factormoduleovertopping ::calculatetanalpha
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
geometrymoduleovertopping ::checkcrosssection
geometrymoduleovertopping ::initializegeometry
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
dllovertopping::calculateqof
dllovertopping::calculateqo
dllovertopping::validateinputf
dllovertopping::validateinputc
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.4.2.7 subroutine, public geometrymoduleovertopping::checkcrosssection (real(kind=wp), intent(in) *psi*, integer, intent(in) *nCoordinates*, real(kind=wp), dimension (ncoordinates), intent(in) *xCoordinates*, real(kind=wp), dimension (ncoordinates), intent(in) *yCoordinates*, real(kind=wp), dimension(ncoordinates-1), intent(in) *roughnessFactors*, logical, intent(out) *succes*, character(len=∗), intent(out) *errorMessage*)

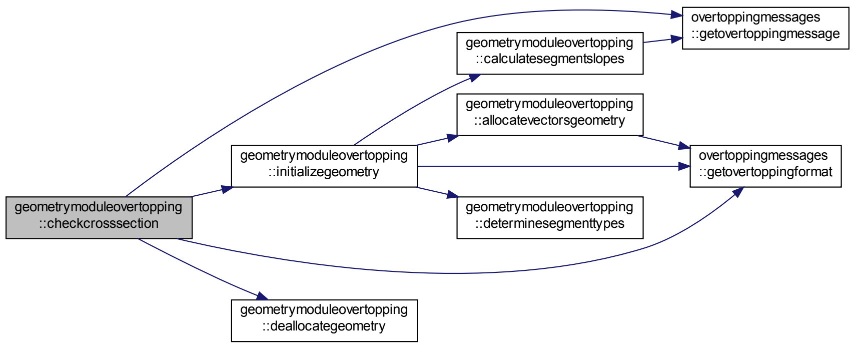
checkCrossSection: check cross section

Parameters

in	*psi*	dike normal (degrees)
in	*ncoordinates*	number of coordinates
in	*xcoordinates*	x-coordinates (m)
in	*ycoordinates*	y-coordinates (m+NAP)
in	*roughnessfac-tors*	roughness factors
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 59 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:


geometrymoduleovertopping ::checkcrosssection
geometrymoduleovertopping ::initializegeometry
geometrymoduleovertopping ::deallocategeometry
geometrymoduleovertopping ::calculatesegmentslopes
geometrymoduleovertopping ::allocatevectorsgeometry
geometrymoduleovertopping ::determinesegmenttypes
overtoppingmessages ::getovertoppingmessage
overtoppingmessages ::getovertoppingformat

4.4.2.8 subroutine, public geometrymoduleovertopping::copygeometry (type (tpgeometry), intent(in) *geometry*, type (tpgeometry), intent(inout) *geometryCopy*, logical, intent(out) *succes*, character(len=∗), intent(inout) *errorMessage*)

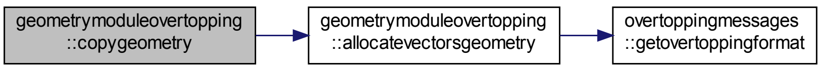
copyGeometry: copy a geometry structure

Parameters

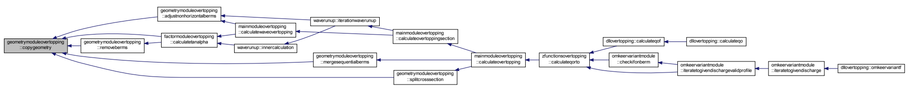
`in`	*geometry*	structure with geometry data
`in,out`	*geometrycopy*	structure with geometry data copy
`out`	*succes*	succes flag
`in,out`	*errormessage*	error message, only set in case of error

Definition at line 390 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:


geometrymoduleovertopping ::copygeometry
geometrymoduleovertopping ::allocatevectorsgeometry
overtoppingmessages ::getovertoppingformat

Here is the caller graph for this function:


geometrymoduleovertopping ::copygeometry
geometrymoduleovertopping ::adjustnonhorizontalberms
geometrymoduleovertopping ::removeberms
factormoduleovertopping ::calculatetanalpha
mainmoduleovertopping ::calculatewaveovertopping
waverunup::iterationwaverunup
waverunup::innercalculation
mainmoduleovertopping ::calculateovertoppingsection
geometrymoduleovertopping ::mergesequentialberms
geometrymoduleovertopping ::splitcrosssection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.4.2.9 subroutine, public geometrymoduleovertopping::deallocategeometry (type (tpgeometry), intent(inout) *geometry*)

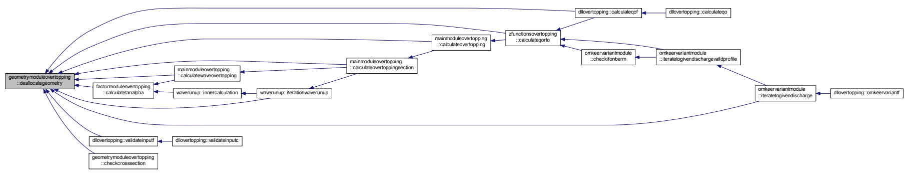
deallocateGeometry: deallocate the geometry vectors

Parameters

`in,out`	*geometry*	structure with geometry data

Definition at line 285 of file geometryModuleOvertopping.f90.

Here is the caller graph for this function:


geometrymoduleovertopping ::deallocategeometry
dllovertopping::calculateqof
dllovertopping::calculateqo
zfunctionsovertopping ::calculateqorto
mainmoduleovertopping ::calculateovertopping
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculatewaveovertopping
factormoduleovertopping ::calculatetanalpha
waverunup::innercalculation
waverunup::iterationwaverunup
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf
dllovertopping::validateinputf
dllovertopping::validateinputc
geometrymoduleovertopping ::checkcrosssection

4.4.2.10 subroutine, public geometrymoduleovertopping::determinesegmenttypes (type (tpgeometry), intent(inout) *geometry*)

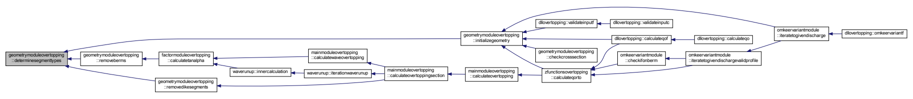
determineSegmentTypes: determine the segment types

Parameters

`in,out`	*geometry*	structure with geometry data

Definition at line 347 of file geometryModuleOvertopping.f90.

Here is the caller graph for this function:



4.4.2.11 subroutine, public geometrymoduleovertopping::initializegeometry (real(kind=wp), intent(in) *psi,* integer, intent(in) *nCoordinates,* real(kind=wp), dimension (ncoordinates), intent(in) *xCoordinates,* real(kind=wp), dimension (ncoordinates), intent(in) *yCoordinates,* real(kind=wp), dimension(ncoordinates-1), intent(in) *roughnessFactors,* type (tpgeometry), intent(out) *geometry,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

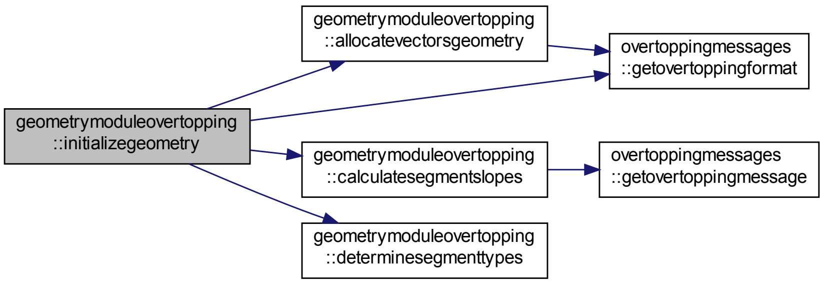
initializeGeometry: initialize the geometry

Parameters

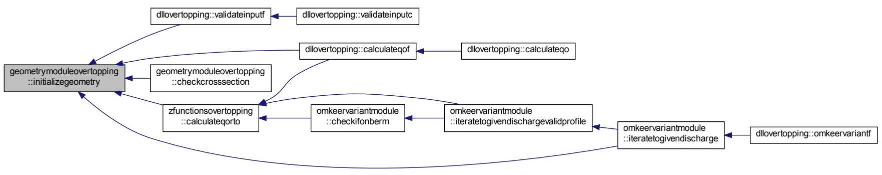
`in`	*psi*	dike normal (degree)
`in`	*ncoordinates*	number of coordinates
`in`	*xcoordinates*	x-coordinates (m)
`in`	*ycoordinates*	y-coordinates (m+NAP)
`in`	*roughnessfactors*	roughness factors
`out`	*geometry*	structure with geometry data
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 170 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:


geometrymoduleovertopping ::initializegeometry
geometrymoduleovertopping ::allocatevectorsgeometry
overtoppingmessages ::getovertoppingformat
geometrymoduleovertopping ::calculatesegmentslopes
overtoppingmessages ::getovertoppingmessage
geometrymoduleovertopping ::determinesegmenttypes

Here is the caller graph for this function:


geometrymoduleovertopping ::initializegeometry
dllovertopping::validateinputf
dllovertopping::validateinputc
dllovertopping::calculateqof
dllovertopping::calculateqo
geometrymoduleovertopping ::checkcrosssection
zfunctionsovertopping ::calculateqorto
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.4.2.12 subroutine, public geometrymoduleovertopping::mergesequentialberms (type (tpgeometry), intent(in) *geometry,* type (tpgeometry), intent(inout) *geometryMergedBerms,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

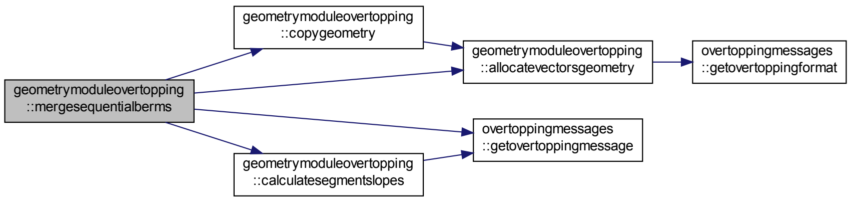
mergeSequentialBerms: merge sequential berms

Parameters

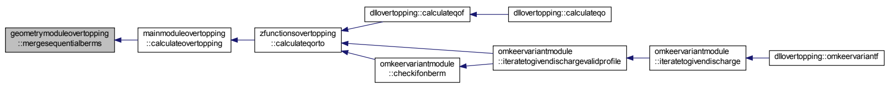
in	*geometry*	structure with geometry data
in,out	*geome-trymergedberms*	geometry data with merged sequential berms
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 441 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:


geometrymoduleovertopping ::mergesequentialberms
geometrymoduleovertopping ::copygeometry
geometrymoduleovertopping ::allocatevectorsgeometry
overtoppingmessages ::getovertoppingformat
overtoppingmessages ::getovertoppingmessage
geometrymoduleovertopping ::calculatesegmentslopes

Here is the caller graph for this function:


geometrymoduleovertopping ::mergesequentialberms
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.4.2.13 subroutine, public geometrymoduleovertopping::removeberms (type (tpgeometry), intent(in) *geometry,* type (tpgeometry), intent(out) *geometryNoBerms,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

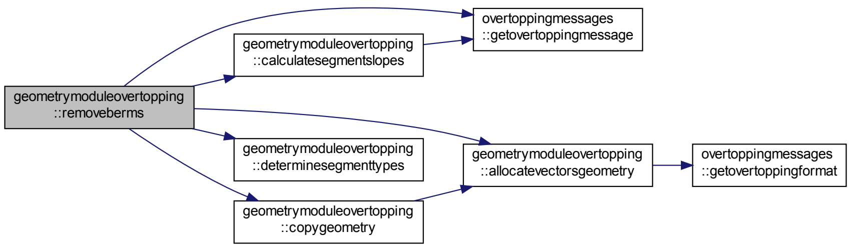
removeBerms: remove berms

Parameters

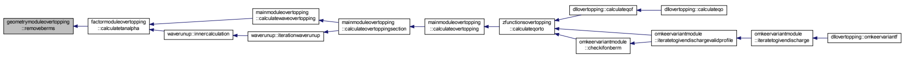
in	*geometry*	structure with geometry data
out	*geome-trynoberms*	geometry data withouth berms
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 639 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:


geometrymoduleovertopping ::removeberms
overtoppingmessages ::getovertoppingmessage
geometrymoduleovertopping ::calculatesegmentslopes
geometrymoduleovertopping ::determinesegmenttypes
geometrymoduleovertopping ::allocatevectorsgeometry
overtoppingmessages ::getovertoppingformat
geometrymoduleovertopping ::copygeometry

Here is the caller graph for this function:


geometrymoduleovertopping ::removeberms
factormoduleovertopping ::calculatetanalpha
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.4.2.14 subroutine, public geometrymoduleovertopping::removedikesegments (type (tpgeometry), intent(in) *geometry,* integer, intent(in) *index,* type (tpgeometry), intent(out) *geometryAdjusted,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

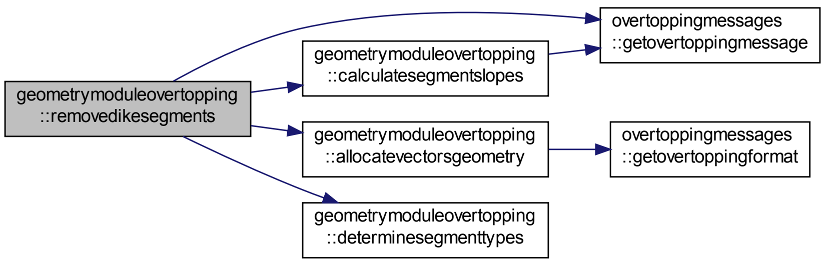
removeDikeSegments: remove dike segments

Parameters

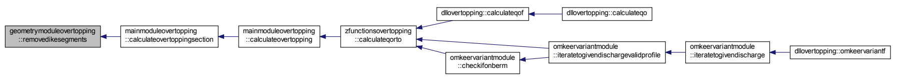
`in`	*geometry*	structure with geometry data
`in`	*index*	index starting point new cross section
`out`	*geometryad-justed*	geometry data with removed dike segments
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 739 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:


geometrymoduleovertopping ::removedikesegments
geometrymoduleovertopping ::calculatesegmentslopes
geometrymoduleovertopping ::allocatevectorsgeometry
geometrymoduleovertopping ::determinesegmenttypes
overtoppingmessages ::getovertoppingmessage
overtoppingmessages ::getovertoppingformat

Here is the caller graph for this function:


geometrymoduleovertopping ::removedikesegments
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.4.2.15 subroutine, public geometrymoduleovertopping::splitcrosssection (type (tpgeometry), intent(in) *geometry,* real(kind=wp), intent(in) *L0,* integer, intent(out) *NwideBerms,* type (tpgeometry), intent(out) *geometrysectionB,* type (tpgeometry), intent(out) *geometrysectionF,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

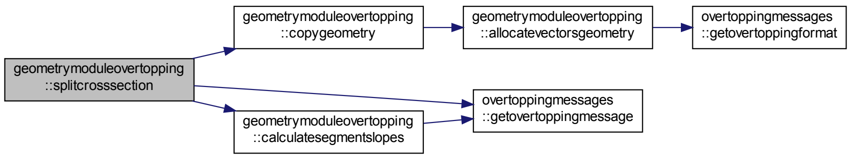
splitCrossSection: split a cross section

Parameters

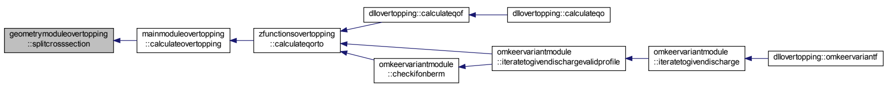
in	*geometry*	structure with geometry data
in	*l0*	wave length (m)
out	*nwideberms*	number of wide berms
out	*geometrysec-tionb*	geometry data with wide berms to ordinary berms
out	*geometrysec-tionf*	geometry data with wide berms to foreshores
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 803 of file geometryModuleOvertopping.f90.

Here is the call graph for this function:


geometrymoduleovertopping ::splitcrosssection
geometrymoduleovertopping ::copygeometry
geometrymoduleovertopping ::allocatevectorsgeometry
overtoppingmessages ::getovertoppingformat
overtoppingmessages ::getovertoppingmessage
geometrymoduleovertopping ::calculatesegmentslopes

Here is the caller graph for this function:


geometrymoduleovertopping ::splitcrosssection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.5 mainmoduleovertopping Module Reference

core computations for Dikes Overtopping

Functions/Subroutines

- subroutine, public calculateovertopping (geometry, load, modelFactors, overtopping, succes, errorMessage)

  calculateOvertopping: calculate the overtopping
- subroutine, public calculateovertoppingsection (geometry, h, Hm0, Tm_10, L0, gammaBeta_z, gammaBeta↩_o, modelFactors, overtopping, succes, errorMessage)

  calculateOvertoppingSection: calculate the overtopping for a section
- subroutine, public calculatewaveovertopping (geometry, h, Hm0, Tm_10, z2, gammaBeta_o, modelFactors, Qo, succes, errorMessage)

  calculateWaveOvertopping: calculate wave overtopping
- subroutine calculateovertoppingnegativefreeboard (load, geometry, overtopping, succes, errorMessage)

  calculateOvertoppingNegativeFreeboard: calculate overtopping in case of negative freeboard
- subroutine, public interpolateresultssections (geometry, L0, NwideBerms, overtoppingB, overtoppingF, over-topping, succes, errorMessage)

  interpolateResultsSections: interpolate results for split cross sections
- subroutine, public checkinputdata (geometry, load, modelFactors, succes, errorMessage)

  checkInputdata: check the input data

- subroutine, public checkmodelfactors (modelFactors, dimErrMessage, errorMessages, ierr)
  checkModelFactors: check the input data

4.5.1 Detailed Description

core computations for Dikes Overtopping

4.5.2 Function/Subroutine Documentation

4.5.2.1 subroutine, public mainmoduleovertopping::calculateovertopping (type (tpgeometry), intent(in) *geometry,* type (tpload), intent(in) *load,* type (tpovertoppinginput), intent(in) *modelFactors,* type (tpovertopping), intent(out) *overtopping,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

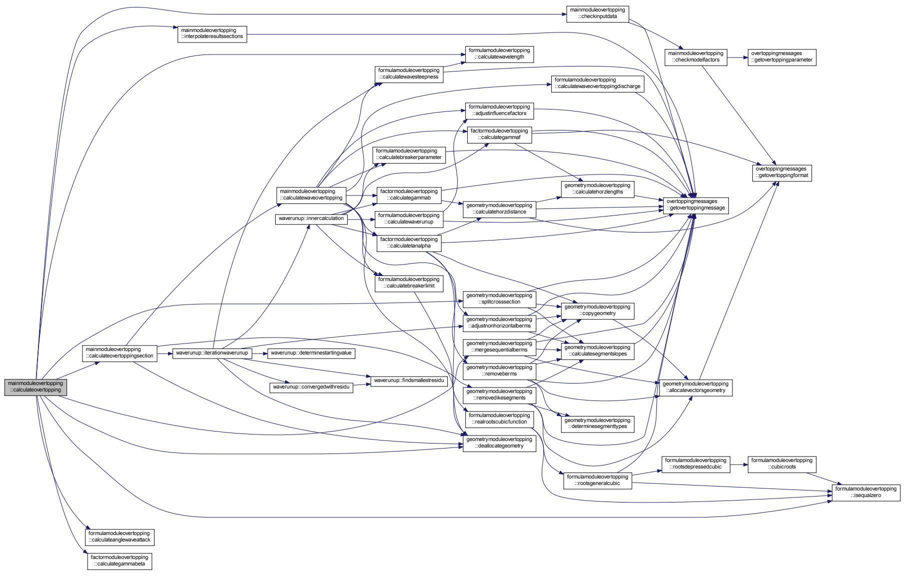
calculateOvertopping: calculate the overtopping

Parameters

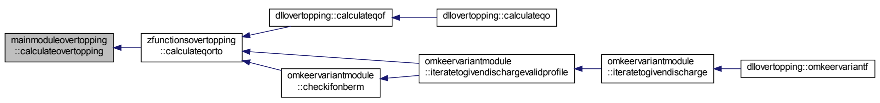
in	*geometry*	structure with geometry data
in	*load*	structure with load parameters
in	*modelfactors*	structure with model factors
out	*overtopping*	structure with overtopping results
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 60 of file mainModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:


mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.5.2.2 subroutine mainmoduleovertopping::calculateovertoppingnegativefreeboard (type (tpload), intent(in) *load,* type (tpgeometry), intent(in) *geometry,* type (tpovertopping), intent(inout) *overtopping,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*) `[private]`

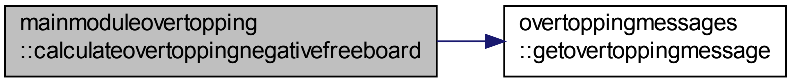
calculateOvertoppingNegativeFreeboard: calculate overtopping in case of negative freeboard

Parameters

`in`	*geometry*	structure with geometry data
`in`	*load*	structure with load parameters
`in,out`	*overtopping*	structure with overtopping results
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 502 of file mainModuleOvertopping.f90.

Here is the call graph for this function:



4.5.2.3 subroutine, public mainmoduleovertopping::calculateovertoppingsection (type (tpgeometry), intent(in) *geometry,* real(kind=wp), intent(in) *h,* real(kind=wp), intent(in) *Hm0,* real(kind=wp), intent(in) *Tm_10,* real(kind=wp), intent(in) *L0,* real(kind=wp), intent(inout) *gammaBeta_z,* real(kind=wp), intent(inout) *gammaBeta_o,* type (tpovertoppinginput), intent(in) *modelFactors,* type (tpovertopping), intent(out) *overtopping,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

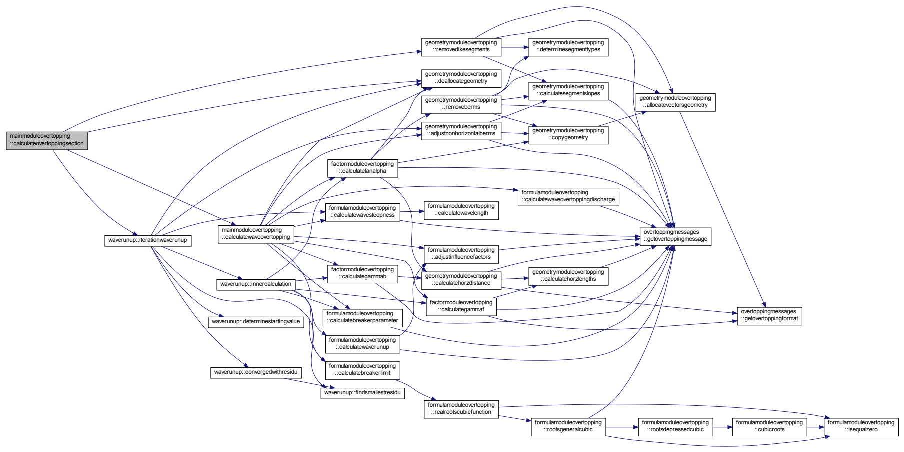
calculateOvertoppingSection: calculate the overtopping for a section

Parameters

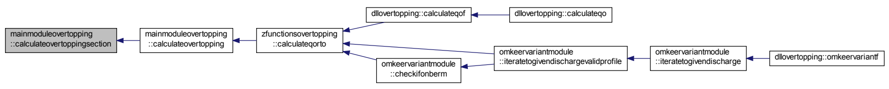
`in`	*geometry*	structure with geometry data
`in`	*h*	local water level (m+NAP)
`in`	*hm0*	significant wave height (m)
`in`	*tm_10*	spectral wave period (s)
`in`	*l0*	wave length (m)
`in,out`	*gammabeta_z*	influence angle wave attack wave run-up
`in,out`	*gammabeta_o*	influence angle wave attack overtopping
`in`	*modelfactors*	structure with model factors
`out`	*overtopping*	structure with overtopping results
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 189 of file mainModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:


mainmoduleovertopping
::calculateovertoppingsection
mainmoduleovertopping
::calculateovertopping
zfunctionsovertopping
::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule
::checkifonberm
omkeervariantmodule
::iteratetogivendischargevalidprofile
omkeervariantmodule
::iteratetogivendischarge
dllovertopping::omkeervariantf

4.5.2.4 subroutine, public mainmoduleovertopping::calculatewaveovertopping (type (tpgeometry), intent(in) *geometry,* real(kind=wp), intent(in) *h,* real(kind=wp), intent(in) *Hm0,* real(kind=wp), intent(in) *Tm_10,* real(kind=wp), intent(in) *z2,* real(kind=wp), intent(inout) *gammaBeta_o,* type (tpovertoppinginput), intent(in) *modelFactors,* real(kind=wp), intent(out) *Qo,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

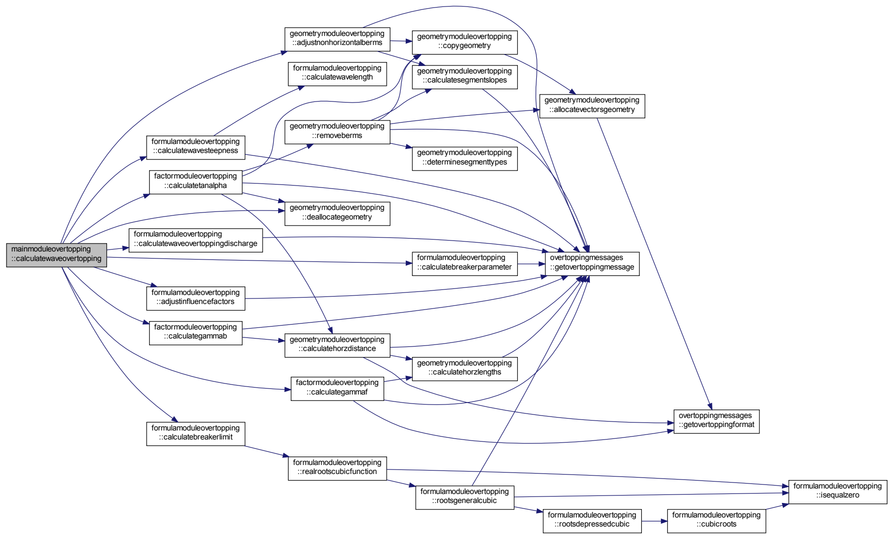
calculateWaveOvertopping: calculate wave overtopping

Parameters

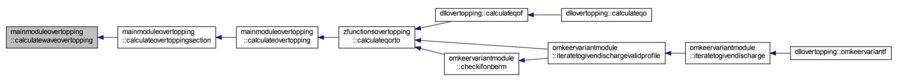
`in`	*geometry*	structure with geometry data
`in`	*h*	local water level (m+NAP)
`in`	*hm0*	significant wave height (m)
`in`	*tm_10*	spectral wave period (s)
`in`	*z2*	2% wave run-up (m)
`in,out`	*gammabeta_o*	influence angle wave attack overtopping
`in`	*modelfactors*	structure with model factors
`out`	*qo*	wave overtopping discharge (m3/m per s)
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 415 of file mainModuleOvertopping.f90.

Here is the call graph for this function:


mainmoduleovertopping ::calculatewaveovertopping
geometrymoduleovertopping ::adjustnonhorizontalberms
geometrymoduleovertopping ::copygeometry
formulamoduleovertopping ::calculatewavelength
geometrymoduleovertopping ::calculatesegmentslopes
geometrymoduleovertopping ::allocatevectorsgeometry
geometrymoduleovertopping ::removeberms
formulamoduleovertopping ::calculatewavesteepness
geometrymoduleovertopping ::determinesegmenttypes
factormoduleovertopping ::calculatetanalpha
geometrymoduleovertopping ::deallocategeometry
formulamoduleovertopping ::calculatewaveovertoppingdischarge
formulamoduleovertopping ::calculatebreakerparameter
overtoppingmessages ::getovertoppingmessage
formulamoduleovertopping ::adjustinfluencefactors
factormoduleovertopping ::calculategammab
geometrymoduleovertopping ::calculatehorzdistance
geometrymoduleovertopping ::calculatehorzlengths
factormoduleovertopping ::calculategammaf
overtoppingmessages ::getovertoppingformat
formulamoduleovertopping ::calculatebreakerlimit
formulamoduleovertopping ::realrootscubicfunction
formulamoduleovertopping ::isequalzero
formulamoduleovertopping ::rootsgeneralcubic
formulamoduleovertopping ::rootsdepressedcubic
formulamoduleovertopping ::cubicroots

Here is the caller graph for this function:


mainmoduleovertopping ::calculatewaveovertopping
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.5.2.5 subroutine, public mainmoduleovertopping::checkinputdata (type (tpgeometry), intent(in) *geometry,* type (tpload), intent(in) *load,* type (tpovertoppinginput), intent(in) *modelFactors,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

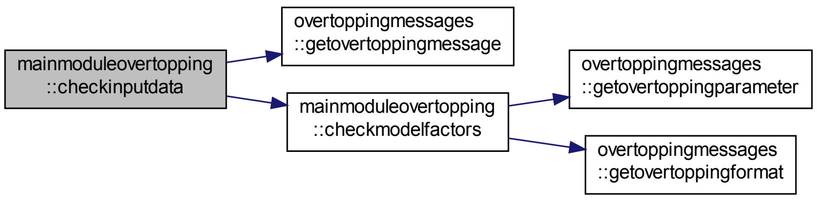
checkInputdata: check the input data

Parameters

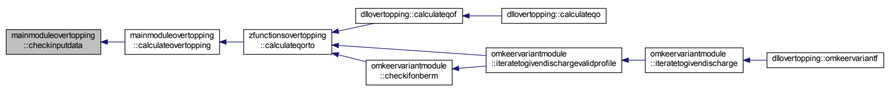
in	*geometry*	structure with geometry data
in	*load*	structure with load parameters
in	*modelfactors*	structure with model factors
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 619 of file mainModuleOvertopping.f90.

Here is the call graph for this function:


mainmoduleovertopping ::checkinputdata
overtoppingmessages ::getovertoppingmessage
mainmoduleovertopping ::checkmodelfactors
overtoppingmessages ::getovertoppingparameter
overtoppingmessages ::getovertoppingformat

Here is the caller graph for this function:


mainmoduleovertopping ::checkinputdata
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.5.2.6 subroutine, public mainmoduleovertopping::checkmodelfactors (type (tpovertoppinginput), intent(in) *modelFactors,* integer, intent(in) *dimErrMessage,* character(len=∗), dimension(dimerrmessage), intent(out) *errorMessages,* integer, intent(out) *ierr*)

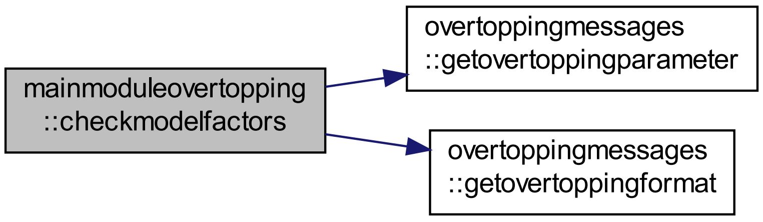
checkModelFactors: check the input data

Parameters

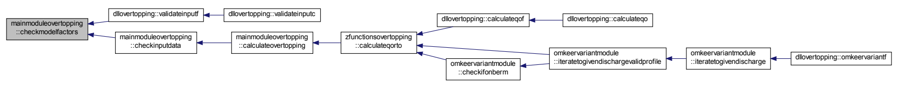
in	*modelfactors*	structure with model factors
in	*dimerrmessage*	max. number of error messages
out	*ierr*	number of errors found
out	*errormessages*	error message

Definition at line 681 of file mainModuleOvertopping.f90.

Here is the call graph for this function:


mainmoduleovertopping ::checkmodelfactors
overtoppingmessages ::getovertoppingparameter
overtoppingmessages ::getovertoppingformat

Here is the caller graph for this function:


mainmoduleovertopping ::checkmodelfactors
dllovertopping::validateinputf
dllovertopping::validateinputc
mainmoduleovertopping ::checkinputdata
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.5.2.7 subroutine, public mainmoduleovertopping::interpolateresultssections (type (tpgeometry), intent(in) *geometry,* real(kind=wp), intent(in) *L0,* integer, intent(in) *NwideBerms,* type (tpovertopping), intent(in) *overtoppingB,* type (tpovertopping), intent(in) *overtoppingF,* type (tpovertopping), intent(out) *overtopping,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

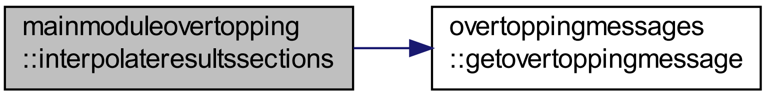
interpolateResultsSections: interpolate results for split cross sections

Parameters

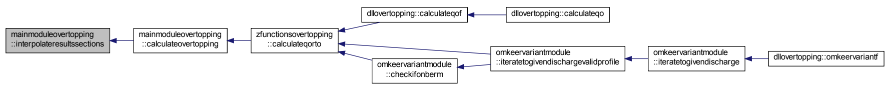
`in`	*geometry*	structure with geometry data
`in`	*l0*	wave length (m)
`in`	*nwideberms*	number of wide berms
`in`	*overtoppingb*	structure with overtopping results ordinary berms
`in`	*overtoppingf*	structure with overtopping results foreshores
`out`	*overtopping*	structure with combined overtopping results
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 538 of file mainModuleOvertopping.f90.

Here is the call graph for this function:



Here is the caller graph for this function:


mainmoduleovertopping
::interpolateresultssections
mainmoduleovertopping
::calculateovertopping
zfunctionsovertopping
::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule
::checkifonberm
omkeervariantmodule
::iteratetogivendischargevalidprofile
omkeervariantmodule
::iteratetogivendischarge
dllovertopping::omkeervariantf

4.6 modulelogging Module Reference

steering the extra logging

Data Types

- type tlogging

  TLogging: structure for steering the logging.

Variables

- integer, parameter maxfilenamelength = 256

  maximum length of filename
- type(tlogging) currentlogging

  copy of argument logging

4.6.1 Detailed Description

steering the extra logging

4.6.2 Variable Documentation

4.6.2.1 type(tlogging) modulelogging::currentlogging

copy of argument logging

Definition at line 43 of file ModuleLogging.f90.

4.6.2.2 integer, parameter modulelogging::maxfilenamelength = 256

maximum length of filename

Definition at line 35 of file ModuleLogging.f90.

4.7 omkeervariantmodule Module Reference

Module for the 'omkeerVariant'.

Functions/Subroutines

- subroutine, public iteratetogivendischarge (load, geometryF, givenDischarge, dikeHeight, modelFactors, overtopping, success, errorText, logging)

    Subroutine with omkeerVariant.
- subroutine iteratetogivendischargevalidprofile (load, geometry, givenDischarge, dikeHeight, modelFactors, overtopping, success, errorText)

    Subroutine with iterateToGivenDischarge, with already checked profile.
- subroutine checkifonberm (geometry, load, modelfactors, overtopping, givenDischarge, dikeHeight, iUp, iLow, dis1, dis2, minDikeHeight, maxDikeHeight, foundValue, success, errorText)

Variables

- real(kind=wp), parameter toldischarge = 1d-3
- real(kind=wp), parameter toldikeheight = 1d-3
- real(kind=wp), parameter minzberm = 0.1_wp
- logical, dimension(:), allocatable isberm
- logical, dimension(:), allocatable isvalidz
- real(kind=wp), dimension(:), allocatable dischargeprofile
- real(kind=wp), dimension(:), allocatable zprofile

4.7.1 Detailed Description

Module for the 'omkeerVariant'.

4.7.2 Function/Subroutine Documentation

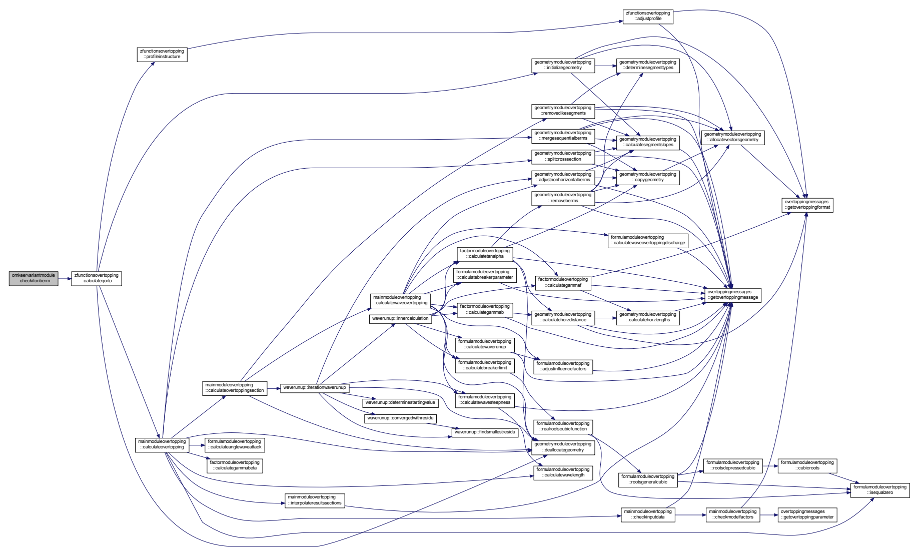
4.7.2.1 subroutine omkeervariantmodule::checkifonberm (type(tpgeometry), intent(in) *geometry,* type(tpload), intent(in) *load,* type(tpovertoppinginput), intent(inout) *modelfactors,* type(tpovertopping), intent(inout) *overtopping,* real(kind=wp), intent(in) *givenDischarge,* real(kind=wp), intent(inout) *dikeHeight,* integer, intent(in) *iUp,* integer, intent(in) *iLow,* real(kind=wp), intent(inout) *dis1,* real(kind=wp), intent(inout) *dis2,* real(kind=wp), intent(inout) *minDikeHeight,* real(kind=wp), intent(inout) *maxDikeHeight,* logical, intent(inout) *foundValue,* logical, intent(inout) *success,* character(len=∗), intent(inout) *errorText*)

Parameters

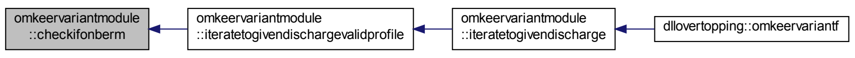
in	*geometry*	internal structure with geometry data
in	*load*	struct with waterlevel and wave parameters
in,out	*modelfactors*	struct with modelFactors
in,out	*overtopping*	structure with overtopping results
in	*iup*	upper bound on profile
in	*ilow*	lower bound on profile
in	*givendischarge*	discharge to iterate to
in,out	*dis1*	discharge at minDikeHeight
in,out	*dis2*	discharge at maxDikeHeight
in,out	*mindikeheight*	lower bound dike heigth
in,out	*maxdikeheight*	upper bound dike heigth
in,out	*dikeheight*	dike height
in,out	*foundvalue*	flag for early succesfull return
in,out	*success*	flag for success
in,out	*errortext*	error message (only set if not successful)

Definition at line 245 of file omkeerVariantModule.f90.

Here is the call graph for this function:



Here is the caller graph for this function:


omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.7.2.2 subroutine, public omkeervariantmodule::iteratetogivendischarge (type(tpload), intent(in) *load,* type(overtoppinggeometrytypef), intent(in) *geometryF,* real(kind=wp), intent(in) *givenDischarge,* real(kind=wp), intent(out) *dikeHeight,* type(tpovertoppinginput), intent(inout) *modelFactors,* type (tpovertopping), intent(inout) *overtopping,* logical, intent(out) *success,* character(len=∗), intent(out) *errorText,* type(tlogging), intent(in) *logging*)

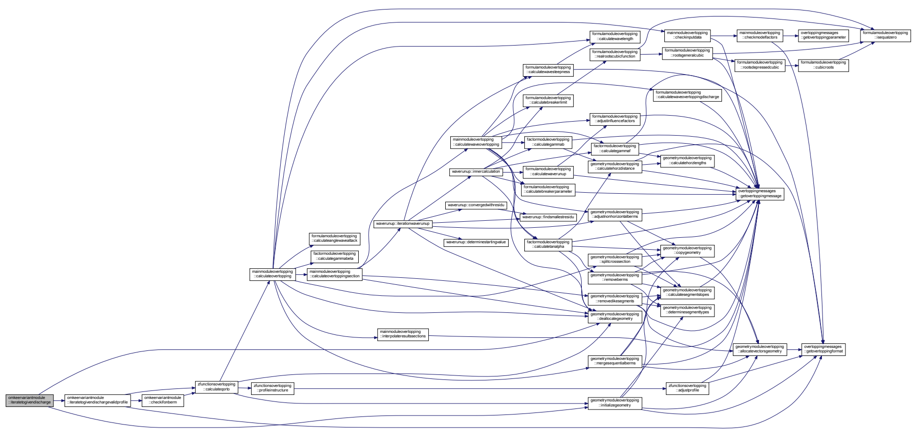
Subroutine with omkeerVariant.

Parameters

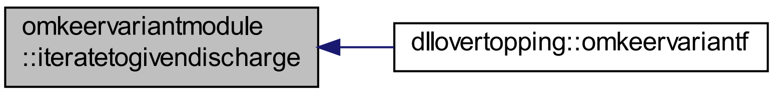
in	*geometryf*	struct with geometry and roughness
in	*load*	struct with waterlevel and wave parameters
in	*givendischarge*	discharge to iterate to
out	*dikeheight*	dike height
in,out	*modelfactors*	struct with modelFactors
in,out	*overtopping*	structure with overtopping results
out	*success*	flag for success
out	*errortext*	error message (only set if not successful)
in	*logging*	logging struct

Definition at line 61 of file omkeerVariantModule.f90.

Here is the call graph for this function:



Here is the caller graph for this function:


omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.7.2.3 subroutine omkeervariantmodule::iteratetogivendischargevalidprofile (type(tpload), intent(in) *load,* type(tpgeometry), intent(in) *geometry,* real(kind=wp), intent(in) *givenDischarge,* real(kind=wp), intent(out) *dikeHeight,* type(tpovertoppinginput), intent(inout) *modelFactors,* type (tpovertopping), intent(inout) *overtopping,* logical, intent(out) *success,* character(len=∗), intent(out) *errorText*) [private]

Subroutine with iterateToGivenDischarge, with already checked profile.

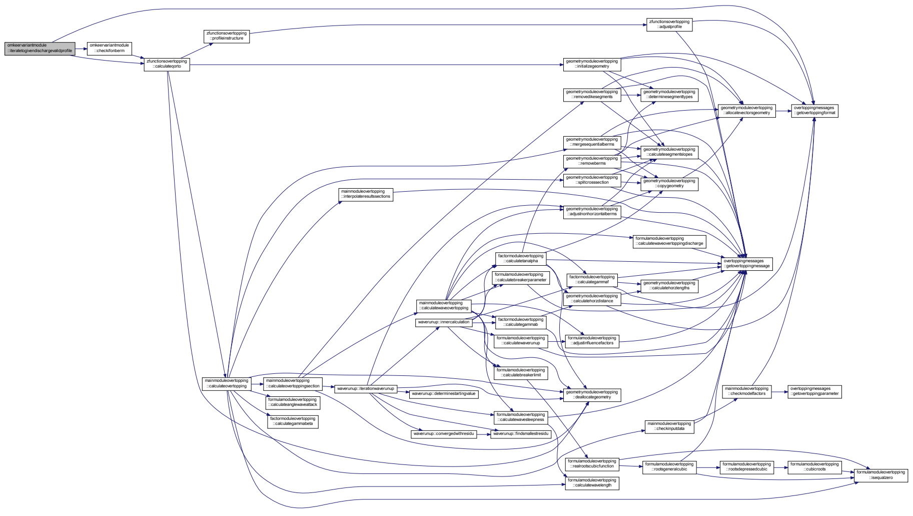
Parameters

in	*geometry*	internal structure with geometry data

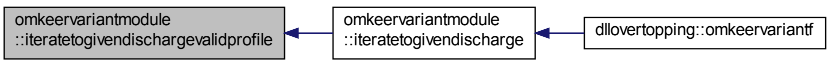
`in`	*load*	struct with waterlevel and wave parameters
`in`	*givendischarge*	discharge to iterate to
`out`	*dikeheight*	dike height
`in,out`	*modelfactors*	struct with modelFactors
`in,out`	*overtopping*	structure with overtopping results
`out`	*success*	flag for success
`out`	*errortext*	error message (only set if not successful)

Definition at line 102 of file omkeerVariantModule.f90.

Here is the call graph for this function:



Here is the caller graph for this function:


omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.7.3 Variable Documentation

4.7.3.1 real(kind=wp), dimension(:), allocatable omkeervariantmodule::dischargeprofile

Definition at line 50 of file omkeerVariantModule.f90.

4.7.3.2 logical, dimension(:), allocatable omkeervariantmodule::isberm

Definition at line 48 of file omkeerVariantModule.f90.

4.7.3.3 logical, dimension(:), allocatable omkeervariantmodule::isvalidz

Definition at line 49 of file omkeerVariantModule.f90.

4.7.3.4 real(kind=wp), parameter omkeervariantmodule::minzberm = 0.1_wp

Definition at line 46 of file omkeerVariantModule.f90.

4.7.3.5 real(kind=wp), parameter omkeervariantmodule::toldikeheight = 1d-3

Definition at line 45 of file omkeerVariantModule.f90.

4.7.3.6 real(kind=wp), parameter omkeervariantmodule::toldischarge = 1d-3

Definition at line 44 of file omkeerVariantModule.f90.

4.7.3.7 real(kind=wp), dimension(:), allocatable omkeervariantmodule::zprofile

Definition at line 51 of file omkeerVariantModule.f90.

4.8 overtoppinginterface Module Reference

Module for the interface of dllOvertopping.

Data Types

- type overtoppinggeometrytype
- type overtoppinggeometrytypef
- type tpprofilecoordinate

Variables

- integer, parameter, public varmodelfactorcriticalovertopping = 8

  Model factor critical overtopping.

4.8.1 Detailed Description

Module for the interface of dllOvertopping.

4.8.2 Variable Documentation

4.8.2.1 integer, parameter, public overtoppinginterface::varmodelfactorcriticalovertopping = 8

Model factor critical overtopping.

Definition at line 38 of file overtoppingInterface.f90.

4.9 overtoppingmessages Module Reference

Module for the messages in the overtopping dll, in Dutch or English.

Functions/Subroutines

- subroutine setlanguage (lang)
  IDs for the strings in this module:
- subroutine getlanguage (lang)
  Subroutine that gets the language for error and validation messages.
- character(len=maxmsg) function getovertoppingmessage (ID)
  Subroutine that returns a message with the corresponding ID in the current language.
- character(len=maxmsg) function getovertoppingformat (ID)
  Subroutine that returns a Fortran format string with the corresponding ID in the current language.
- character(len=maxpar) function getovertoppingparameter (ID)
  Subroutine that returns the name of an input parameter with the corresponding ID in the current language.

Variables

- integer, parameter, private maxmsg = 128
- integer, parameter, private maxpar =32
- character(len=2), private language = 'NL'
  default : Dutch

4.9.1 Detailed Description

Module for the messages in the overtopping dll, in Dutch or English.

4.9.2 Function/Subroutine Documentation

4.9.2.1 subroutine overtoppingmessages::getlanguage (character(len=∗), intent(out) *lang*)

Subroutine that gets the language for error and validation messages.

Parameters

out	*lang*	filled with current language ID

Definition at line 121 of file OvertoppingMessages.f90.

4.9.2.2 character(len=maxmsg) function overtoppingmessages::getovertoppingformat (integer, intent(in) *ID*)

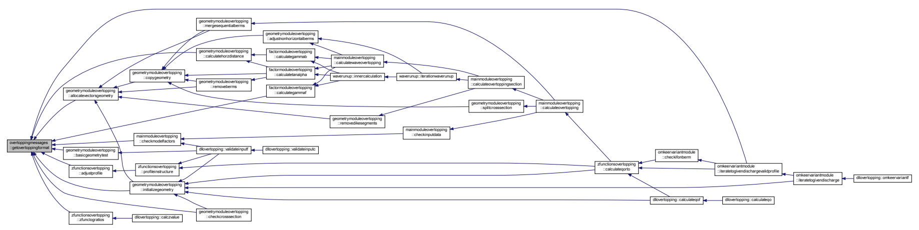
Subroutine that returns a Fortran format string with the corresponding ID in the current language.

Parameters

in	*id*	identification number of string

Definition at line 281 of file OvertoppingMessages.f90.

Here is the caller graph for this function:


overtoppingmessages::getovertoppingformat
geometrymoduleovertopping::mergesequentialberms
geometrymoduleovertopping::adjustnonhorizontalberms
geometrymoduleovertopping::calculatehorzdistance
factormoduleovertopping::calculategammab
mainmoduleovertopping::calculatewaveovertopping
geometrymoduleovertopping::copygeometry
factormoduleovertopping::calculatetanalpha
waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping::calculateovertoppingsection
geometrymoduleovertopping::removeberms
geometrymoduleovertopping::allocatevectorsgeometry
factormoduleovertopping::calculategammaf
geometrymoduleovertopping::splitcrosssection
mainmoduleovertopping::calculateovertopping
geometrymoduleovertopping::removedikesegments
mainmoduleovertopping::checkinputdata
mainmoduleovertopping::checkmodelfactors
geometrymoduleovertopping::basicgeometrytest
dllovertopping::validateinputf
dllovertopping::validateinputc
zfunctionsovertopping::adjustprofile
zfunctionsovertopping::profileinstructure
geometrymoduleovertopping::initializegeometry
zfunctionsovertopping::calculateqorto
omkeervariantmodule::checkifonberm
omkeervariantmodule::iteratetogivendischargevalidprofile
omkeervariantmodule::iteratetogivendischarge
dllovertopping::omkeervariantf
dllovertopping::calculateqof
dllovertopping::calculateqo
geometrymoduleovertopping::checkcrosssection
zfunctionsovertopping::zfunclogratios
dllovertopping::calczvalue

4.9.2.3 character(len=maxmsg) function overtoppingmessages::getovertoppingmessage (integer, intent(in) *ID*)

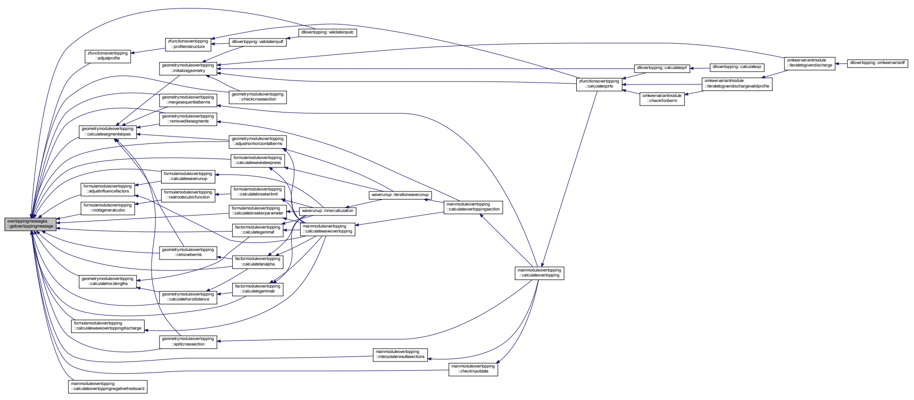
Subroutine that returns a message with the corresponding ID in the current language.

Parameters

in	*id*	identification number of string

Definition at line 131 of file OvertoppingMessages.f90.

Here is the caller graph for this function:



4.9.2.4 character(len=maxpar) function overtoppingmessages::getovertoppingparameter (integer, intent(in) *ID*)

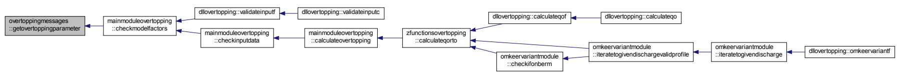
Subroutine that returns the name of an input parameter with the corresponding ID in the current language.

Parameters

in	*id*	identification number of string

Definition at line 339 of file OvertoppingMessages.f90.

Here is the caller graph for this function:


overtoppingmessages ::getovertoppingparameter
mainmoduleovertopping ::checkmodelfactors
dllovertopping::validateinputf
dllovertopping::validateinputc
mainmoduleovertopping ::checkinputdata
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkforberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.9.2.5 subroutine overtoppingmessages::setlanguage (character(len=∗), intent(in) *lang*)

IDs for the strings in this module:

Subroutine that sets the language for error and validation messages only strings 'NL' and 'UK' are recoqnized (lower and upper case)

Parameters

in	*lang*	new language ID to be used

Definition at line 103 of file OvertoppingMessages.f90.

4.9.3 Variable Documentation

4.9.3.1 character(len=2), private overtoppingmessages::language = 'NL'

default : Dutch

Definition at line 38 of file OvertoppingMessages.f90.

4.9.3.2 integer, parameter, private overtoppingmessages::maxmsg = 128

Definition at line 36 of file OvertoppingMessages.f90.

4.9.3.3 integer, parameter, private overtoppingmessages::maxpar =32

Definition at line 36 of file OvertoppingMessages.f90.

4.10 typedefinitionsovertopping Module Reference

type definitions for Dikes Overtopping

Data Types

- type tpgeometry

  tpGeometry: structure with geometry data
- type tpload

  tpLoad: structure with load parameters
- type tpovertopping

  tpOvertopping: structure with overtopping results
- type tpovertoppinginput

  OvertoppingModelFactors: C-structure with model factors.

Variables

- real(kind=wp), parameter frunup1 = 1.65_wp
- real(kind=wp), parameter frunup2 = 4.00_wp
- real(kind=wp), parameter frunup3 = 1.50_wp
- real(kind=wp), parameter xdiff_min = 2.0d-2

  minimal value distance between x-coordinates (m)
- real(kind=wp), parameter margindiff = 1.0d-14

  margin for minimal distance (m)
- real(kind=wp), parameter berm_min = 0.0d0

  minimal value gradient berm segment
- real(kind=wp), parameter berm_max = 1.0d0/15

  maximal value gradient berm segment
- real(kind=wp), parameter slope_min = 1.0d0/8

  minimal value gradient slope segment
- real(kind=wp), parameter slope_max = 1.0d0

  maximal value gradient slope segment
- real(kind=wp), parameter margingrad = 0.0025d0

  margin for minimal and maximal gradients
- real(kind=wp), parameter rfactor_min = 0.5d0

  minimal value roughness factor dike segments

- real(kind=wp), parameter rfactor_max = 1.0d0

  maximal value roughness factor dike segments
- real(kind=wp), parameter mz2_min = 0.0d0

  minimal value model factor of 2% runup height
- real(kind=wp), parameter mz2_max = huge(mz2_max)

  maximal value model factor of 2% runup height
- real(kind=wp), parameter fb_min = 0.0d0

  minimal value model factor for breaking waves
- real(kind=wp), parameter fb_max = huge(fB_max)

  maximal value model factor for breaking waves
- real(kind=wp), parameter fn_min = 0.0d0

  minimal value model factor for non-breaking waves
- real(kind=wp), parameter fn_max = huge(fN_max)

  maximal value model factor for non-breaking waves
- real(kind=wp), parameter fs_min = 0.0d0

  minimal value model factor for shallow waves
- real(kind=wp), parameter fs_max = huge(fS_max)

  maximal value model factor for shallow waves
- real(kind=wp), parameter foreshore_min = 0.3d0

  minimal value reduction factor foreshore
- real(kind=wp), parameter foreshore_max = 1.0d0

  maximal value reduction factor foreshore
- integer, parameter z2_iter_max1 = 49

  maximal number of iterations for calculation z2 part 1
- integer, parameter z2_iter_max2 = 70

  maximal number of iterations for calculation z2 part 1 & 2
- real(kind=wp), parameter z2_margin = 0.001d0

  margin for convergence criterium calculation z2

4.10.1 Detailed Description

type definitions for Dikes Overtopping

4.10.2 Variable Documentation

4.10.2.1 real(kind=wp), parameter typedefinitionsovertopping::berm_max = 1.0d0/15

maximal value gradient berm segment

Definition at line 91 of file typeDefinitionsOvertopping.f90.

4.10.2.2 real(kind=wp), parameter typedefinitionsovertopping::berm_min = 0.0d0

minimal value gradient berm segment

Definition at line 90 of file typeDefinitionsOvertopping.f90.

4.10.2.3 real(kind=wp), parameter typedefinitionsovertopping::fb_max = huge(fB_max)

maximal value model factor for breaking waves

Definition at line 100 of file typeDefinitionsOvertopping.f90.

4.10.2.4 real(kind=wp), parameter typedefinitionsovertopping::fb_min = 0.0d0

minimal value model factor for breaking waves

Definition at line 99 of file typeDefinitionsOvertopping.f90.

4.10.2.5 real(kind=wp), parameter typedefinitionsovertopping::fn_max = huge(fN_max)

maximal value model factor for non-breaking waves

Definition at line 102 of file typeDefinitionsOvertopping.f90.

4.10.2.6 real(kind=wp), parameter typedefinitionsovertopping::fn_min = 0.0d0

minimal value model factor for non-breaking waves

Definition at line 101 of file typeDefinitionsOvertopping.f90.

4.10.2.7 real(kind=wp), parameter typedefinitionsovertopping::foreshore_max = 1.0d0

maximal value reduction factor foreshore

Definition at line 106 of file typeDefinitionsOvertopping.f90.

4.10.2.8 real(kind=wp), parameter typedefinitionsovertopping::foreshore_min = 0.3d0

minimal value reduction factor foreshore

Definition at line 105 of file typeDefinitionsOvertopping.f90.

4.10.2.9 real(kind=wp), parameter typedefinitionsovertopping::frunup1 = 1.65_wp

Definition at line 63 of file typeDefinitionsOvertopping.f90.

4.10.2.10 real(kind=wp), parameter typedefinitionsovertopping::frunup2 = 4.00_wp

Definition at line 64 of file typeDefinitionsOvertopping.f90.

4.10.2.11 real(kind=wp), parameter typedefinitionsovertopping::frunup3 = 1.50_wp

Definition at line 65 of file typeDefinitionsOvertopping.f90.

4.10.2.12 real(kind=wp), parameter typedefinitionsovertopping::fs_max = huge(fS_max)

maximal value model factor for shallow waves

Definition at line 104 of file typeDefinitionsOvertopping.f90.

4.10.2.13 real(kind=wp), parameter typedefinitionsovertopping::fs_min = 0.0d0

minimal value model factor for shallow waves

Definition at line 103 of file typeDefinitionsOvertopping.f90.

4.10.2.14 real(kind=wp), parameter typedefinitionsovertopping::margindiff = 1.0d-14

margin for minimal distance (m)

Definition at line 89 of file typeDefinitionsOvertopping.f90.

4.10.2.15 real(kind=wp), parameter typedefinitionsovertopping::margingrad = 0.0025d0

margin for minimal and maximal gradients

Definition at line 94 of file typeDefinitionsOvertopping.f90.

4.10.2.16 real(kind=wp), parameter typedefinitionsovertopping::mz2_max = huge(mz2_max)

maximal value model factor of 2% runup height

Definition at line 98 of file typeDefinitionsOvertopping.f90.

4.10.2.17 real(kind=wp), parameter typedefinitionsovertopping::mz2_min = 0.0d0

minimal value model factor of 2% runup height

Definition at line 97 of file typeDefinitionsOvertopping.f90.

4.10.2.18 real(kind=wp), parameter typedefinitionsovertopping::rfactor_max = 1.0d0

maximal value roughness factor dike segments

Definition at line 96 of file typeDefinitionsOvertopping.f90.

4.10.2.19 real(kind=wp), parameter typedefinitionsovertopping::rfactor_min = 0.5d0

minimal value roughness factor dike segments

Definition at line 95 of file typeDefinitionsOvertopping.f90.

4.10.2.20 real(kind=wp), parameter typedefinitionsovertopping::slope_max = 1.0d0

maximal value gradient slope segment

Definition at line 93 of file typeDefinitionsOvertopping.f90.

4.10.2.21 real(kind=wp), parameter typedefinitionsovertopping::slope_min = 1.0d0/8

minimal value gradient slope segment

Definition at line 92 of file typeDefinitionsOvertopping.f90.

4.10.2.22 real(kind=wp), parameter typedefinitionsovertopping::xdiff_min = 2.0d-2

minimal value distance between x-coordinates (m)

Definition at line 88 of file typeDefinitionsOvertopping.f90.

4.10.2.23 integer, parameter typedefinitionsovertopping::z2_iter_max1 = 49

maximal number of iterations for calculation z2 part 1

Definition at line 107 of file typeDefinitionsOvertopping.f90.

4.10.2.24 integer, parameter typedefinitionsovertopping::z2_iter_max2 = 70

maximal number of iterations for calculation z2 part 1 & 2

Definition at line 108 of file typeDefinitionsOvertopping.f90.

4.10.2.25 real(kind=wp), parameter typedefinitionsovertopping::z2_margin = 0.001d0

margin for convergence criterium calculation z2

Definition at line 109 of file typeDefinitionsOvertopping.f90.

4.11 waverunup Module Reference

Iteration procedure for 2% wave runup.

Functions/Subroutines

- subroutine, public iterationwaverunup (geometry, h, Hm0, Tm_10, gammaBeta_z, modelFactors, z2, succes, errorMessage)

  iterationWaveRunup: iteration for the wave runup
- real(kind=wp) function innercalculation (geometry, h, Hm0, gammaBeta_z, modelFactors, z2, s0, geometry↩ FlatBerms, succes, errorMessage)

  innerCalculation: inner calculation for the wave runup
- real(kind=wp) function determinestartingvalue (i, relaxationFactor, z2_start, z2_end, Hm0)

  determineStartingValue: helper function to find a start value for z2
- integer function findsmallestresidu (z2_start, z2_end, n)

  findSmallestResidu: helper function to find the smallest residu
- subroutine convergedwithresidu (z2_start, z2_end)

  convergedWithResidu: helper function to handle convergence with a higher residu than expected if logging is enabled, it writes a small message to the logfile

4.11.1 Detailed Description

Iteration procedure for 2% wave runup.

4.11.2 Function/Subroutine Documentation

4.11.2.1 subroutine waverunup::convergedwithresidu (real(kind=wp), dimension(:), intent(in) *z2_start*, real(kind=wp), dimension(:), intent(inout) *z2_end*) [private]

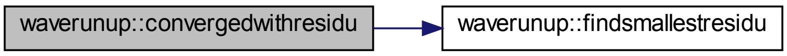
convergedWithResidu: helper function to handle convergence with a higher residu than expected if logging is enabled, it writes a small message to the logfile

Parameters

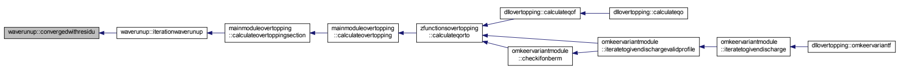
in	*z2_start*	array with z2 values at the start of the iteration i
in,out	*z2_end*	array with z2 values at the end of iteration i

Definition at line 373 of file waveRunup.f90.

Here is the call graph for this function:



Here is the caller graph for this function:


waverunup::convergedwithresidu
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.11.2.2 real(kind=wp) function waverunup::determinestartingvalue (integer, intent(in) *i*, real(kind=wp), intent(in) *relaxationFactor*, real(kind=wp), dimension(:), intent(in) *z2_start*, real(kind=wp), dimension(:), intent(in) *z2_end*, real(kind=wp), intent(in) *Hm0*) `[private]`

determineStartingValue: helper function to find a start value for z2

Parameters

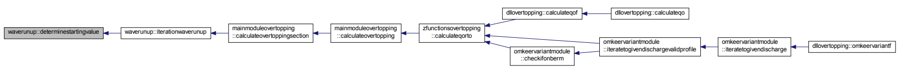
in	*i*	current iteration number
in	*relaxationfactor*	relaxation factor as given by user
in	*z2_start*	array with z2 values at the start of the iteration i
in	*z2_end*	array with z2 values at the end of iteration i
in	*hm0*	significant wave height

Returns

return value: start value for z2 in current iteration

Definition at line 292 of file waveRunup.f90.

Here is the caller graph for this function:


waverunup::determinestartingvalue
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.11.2.3 integer function waverunup::findsmallestresidu (real(kind=wp), dimension(:), intent(in) *z2_start*, real(kind=wp), dimension(:), intent(in) *z2_end*, integer, intent(in), optional *n*) `[private]`

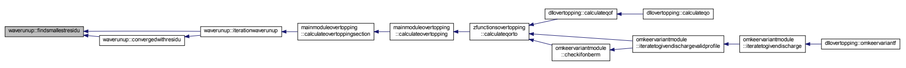
findSmallestResidu: helper function to find the smallest residu

Parameters

in	*z2_start*	array with z2 values at the start of the iteration i
in	*z2_end*	array with z2 values at the end of iteration i
in	*n*	number of iterations already done

Definition at line 330 of file waveRunup.f90.

Here is the caller graph for this function:


waverunup::findsmallestresidu
waverunup::convergedwithresidu
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.11.2.4 real(kind=wp) function waverunup::innercalculation (type (tpgeometry), intent(in) *geometry,* real(kind=wp), intent(in) *h,* real(kind=wp), intent(in) *Hm0,* real(kind=wp), intent(inout) *gammaBeta_z,* type (tpovertoppinginput), intent(in) *modelFactors,* real(kind=wp), intent(in) *z2,* real(kind=wp), intent(in) *s0,* type (tpgeometry), intent(in) *geometryFlatBerms,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*) [private]

innerCalculation: inner calculation for the wave runup

Parameters

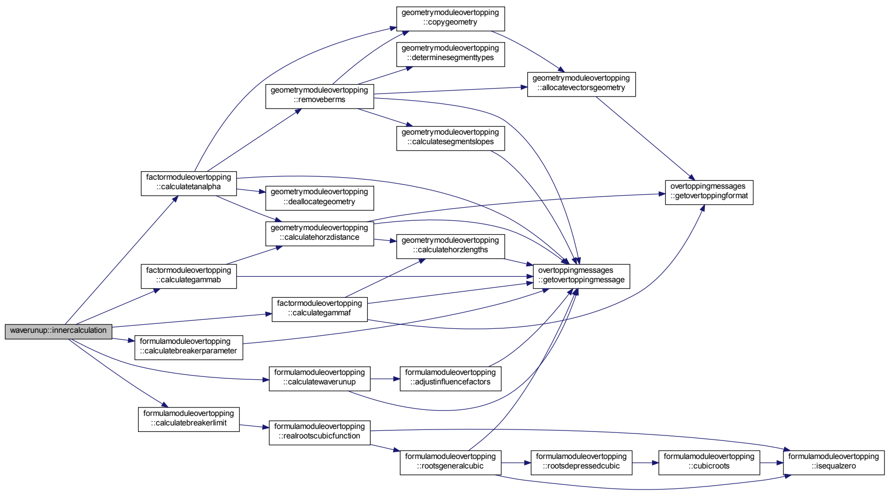
in	*geometry*	structure with geometry data
in	*h*	local water level (m+NAP)
in	*hm0*	significant wave height (m)
in,out	*gammabeta_z*	influence factor angle wave attack 2% run-up
in	*modelfactors*	structure with model factors
in	*z2*	2% wave run-up (m)
in	*s0*	wave steepness
in	*geometryflat-berms*	structure with geometry data with horizontal berms
out	*succes*	flag for succes
out	*errormessage*	error message

Returns

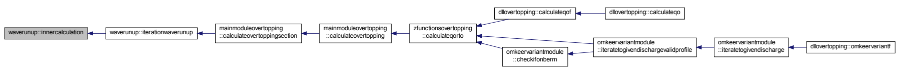
2% wave run-up at end of inner calculation

Definition at line 181 of file waveRunup.f90.

Here is the call graph for this function:


geometrymoduleovertopping ::copygeometry
geometrymoduleovertopping ::determinesegmenttypes
geometrymoduleovertopping ::removeberms
geometrymoduleovertopping ::allocatevectorsgeometry
geometrymoduleovertopping ::calculatesegmentslopes
factormoduleovertopping ::calculatetanalpha
geometrymoduleovertopping ::deallocategeometry
overtoppingmessages ::getovertoppingformat
geometrymoduleovertopping ::calculatehorzdistance
geometrymoduleovertopping ::calculatehorzlengths
factormoduleovertopping ::calculategammab
overtoppingmessages ::getovertoppingmessage
factormoduleovertopping ::calculategammaf
waverunup::innercalculation
formulamoduleovertopping ::calculatebreakerparameter
formulamoduleovertopping ::calculatewaverunup
formulamoduleovertopping ::adjustinfluencefactors
formulamoduleovertopping ::calculatebreakerlimit
formulamoduleovertopping ::realrootscubicfunction
formulamoduleovertopping ::rootsgeneralcubic
formulamoduleovertopping ::rootsdepressedcubic
formulamoduleovertopping ::cubicroots
formulamoduleovertopping ::isequalzero

Here is the caller graph for this function:


waverunup::innercalculation
waverunup::iterationwaverunup
mainmoduleovertopping ::calculateovertoppingsection
mainmoduleovertopping ::calculateovertopping
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.11.2.5 subroutine, public waverunup::iterationwaverunup (type (tpgeometry), intent(in) *geometry,* real(kind=wp), intent(in) *h,* real(kind=wp), intent(in) *Hm0,* real(kind=wp), intent(in) *Tm_10,* real(kind=wp), intent(inout) *gammaBeta_z,* type (tpovertoppinginput), intent(in) *modelFactors,* real(kind=wp), intent(out) *z2,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

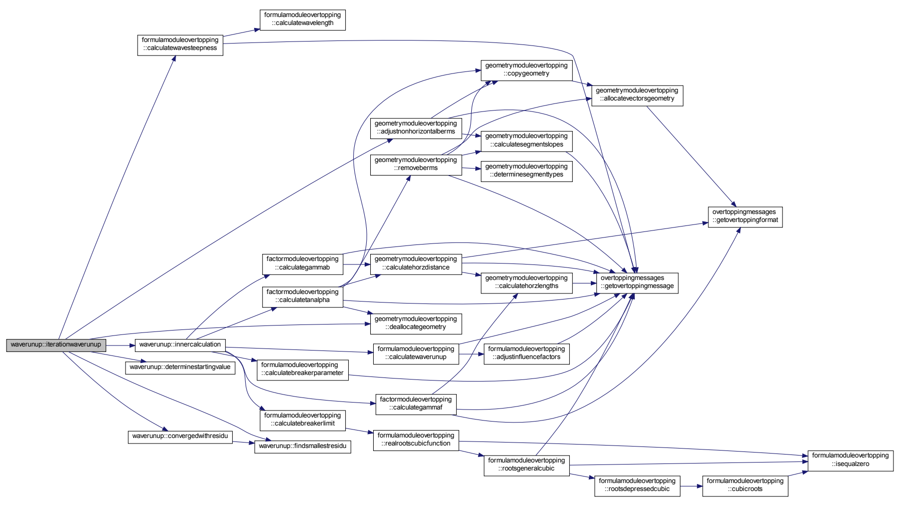
iterationWaveRunup: iteration for the wave runup

Parameters

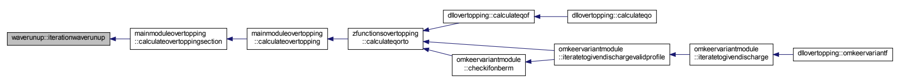
`in`	*geometry*	structure with geometry data
`in`	*h*	local water level (m+NAP)
`in`	*hm0*	significant wave height (m)
`in`	*tm_10*	spectral wave period (s)
`in,out`	*gammabeta_z*	influence factor angle wave attack 2% run-up
`in`	*modelfactors*	structure with model factors
`out`	*z2*	2% wave run-up (m)
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 59 of file waveRunup.f90.

Here is the call graph for this function:



Here is the caller graph for this function:



4.12 zfunctionsovertopping Module Reference

Module for the Limit State Functions (Z-functions) for wave overtopping.

Functions/Subroutines

- subroutine, public calculateqorto (dikeHeight, modelFactors, overtopping, load, geometry, succes, error←Message)

  Subroutine to calculate the overtopping discharge with the Overtopping dll.
- subroutine, public profileinstructure (nrCoordinates, xcoordinates, ycoordinates, dikeHeight, nrCoords←Adjusted, xCoordsAdjusted, zCoordsAdjusted, succes, errorMessage)

  Subroutine to fill the profile in a structure and call the adjustment function of the profile due to a desired dike height.
- subroutine adjustprofile (nrCoordinates, coordinates, dikeHeight, nrCoordsAdjusted, xCoordsAdjusted, z←CoordsAdjusted, succes, errorMessage)

  Subroutine adjust the profile due to a desired dike height.
- real(kind=wp) function, public zfunclogratios (qo, qc, mqo, mqc, success, errorMessage)

  Routine to compute the limit state value by using the logs of the overtopping discharges (computed and desired)

4.12.1 Detailed Description

Module for the Limit State Functions (Z-functions) for wave overtopping.

4.12.2 Function/Subroutine Documentation

4.12.2.1 subroutine zfunctionsovertopping::adjustprofile (integer, intent(in) *nrCoordinates,* type(tpprofilecoordinate), dimension(nrcoordinates), intent(in) *coordinates,* real(kind=wp), intent(in) *dikeHeight,* integer, intent(out) *nrCoordsAdjusted,* real(kind=wp), dimension(:), pointer *xCoordsAdjusted,* real(kind=wp), dimension(:), pointer *zCoordsAdjusted,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*) `[private]`

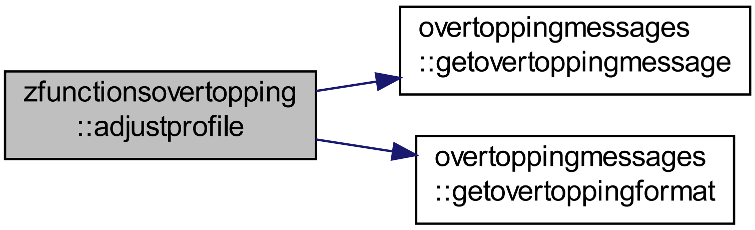
Subroutine adjust the profile due to a desired dike height.

Parameters

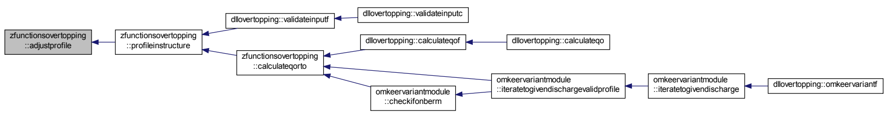
in	*nrcoordinates*	number of coordinates of the profile
in	*coordinates*	structure for the profile
in	*dikeheight*	dike height
out	*nrcoordsad-justed*	number of coordinates in the adjusted profile
	xcoordsadjusted	vector with x-coordinates of the adjusted profile
	zcoordsadjusted	vector with y-coordinates of the adjusted profile
out	*succes*	flag for succes
out	*errormessage*	error message

Definition at line 129 of file zFunctionsOvertopping.f90.

Here is the call graph for this function:


zfunctionsovertopping ::adjustprofile
overtoppingmessages ::getovertoppingmessage
overtoppingmessages ::getovertoppingformat

Here is the caller graph for this function:


zfunctionsovertopping ::adjustprofile
zfunctionsovertopping ::profileinstructure
dllovertopping::validateinputf
dllovertopping::validateinputc
dllovertopping::calculateqof
dllovertopping::calculateqo
zfunctionsovertopping ::calculateqorto
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf
omkeervariantmodule ::checkifonberm

4.12.2.2 subroutine, public zfunctionsovertopping::calculateqorto (real(kind=wp), intent(in) *dikeHeight,* type(tpovertoppinginput), intent(inout) *modelFactors,* type (tpovertopping), intent(out) *overtopping,* type (tpload), intent(in) *load,* type (tpgeometry), intent(in) *geometry,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

Subroutine to calculate the overtopping discharge with the Overtopping dll.

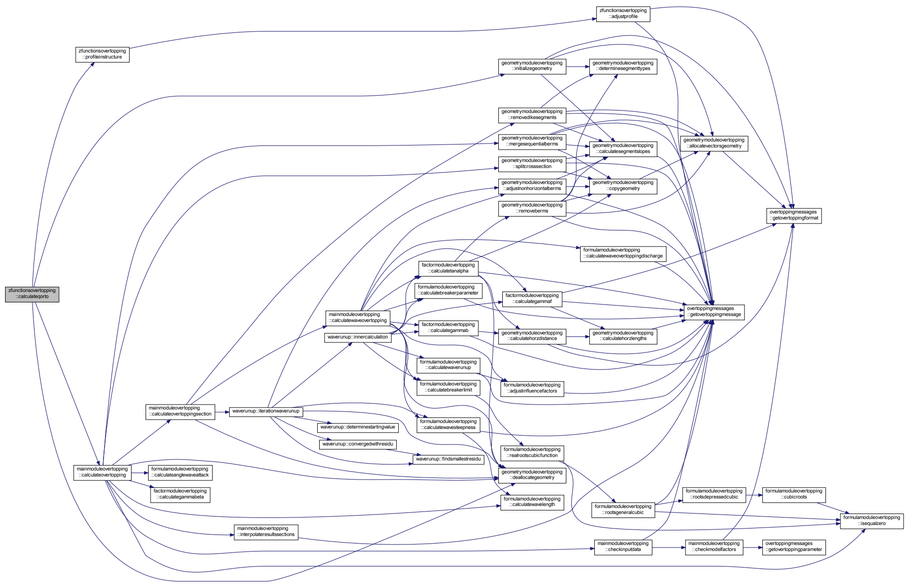
Parameters

in	*dikeheight*	dike height
in,out	*modelfactors*	struct with model factors
out	*overtopping*	structure with overtopping results
in	*geometry*	structure with geometry data
in	*load*	structure with load parameters

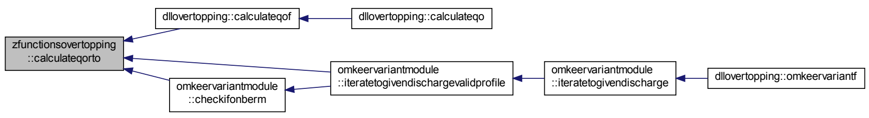
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 55 of file zFunctionsOvertopping.f90.

Here is the call graph for this function:


zfunctionsovertopping ::calculateqorto
zfunctionsovertopping ::profileinstructure
zfunctionsovertopping ::adjustprofile
geometrymoduleovertopping ::initializegeometry
geometrymoduleovertopping ::determinesegmenttypes
geometrymoduleovertopping ::removedikesegments
geometrymoduleovertopping ::mergesequentialberms
geometrymoduleovertopping ::calculatesegmentslopes
geometrymoduleovertopping ::allocatevectorsgeometry
geometrymoduleovertopping ::splitcrosssection
geometrymoduleovertopping ::adjustnonhorizontalberms
geometrymoduleovertopping ::copygeometry
geometrymoduleovertopping ::removeberms
overtoppingmessages ::getovertoppingformat
formulamoduleovertopping ::calculatewaveovertoppingdischarge
factormoduleovertopping ::calculatetanalpha
formulamoduleovertopping ::calculatebreakerparameter
factormoduleovertopping ::calculategammaf
overtoppingmessages ::getovertoppingmessage
mainmoduleovertopping ::calculatewaveovertopping
waverunup::innercalculation
factormoduleovertopping ::calculategammab
geometrymoduleovertopping ::calculatehorzdistance
geometrymoduleovertopping ::calculatehorzlengths
formulamoduleovertopping ::calculatewaverunup
formulamoduleovertopping ::calculatebreakerlimit
formulamoduleovertopping ::adjustinfluencefactors
mainmoduleovertopping ::calculateovertoppingsection
waverunup::iterationwaverunup
waverunup::determinestartingvalue
formulamoduleovertopping ::calculatewavesteepness
waverunup::convergedwithresidu
waverunup::findsmallestresidu
formulamoduleovertopping ::realrootscubicfunction
mainmoduleovertopping ::calculateovertopping
formulamoduleovertopping ::calculateanglewaveattack
geometrymoduleovertopping ::deallocategeometry
factormoduleovertopping ::calculategammabeta
formulamoduleovertopping ::calculatewavelength
formulamoduleovertopping ::rootsgeneralcubic
formulamoduleovertopping ::rootsdepressedcubic
formulamoduleovertopping ::cubicroots
formulamoduleovertopping ::isequalzero
mainmoduleovertopping ::interpolateresultssections
mainmoduleovertopping ::checkinputdata
mainmoduleovertopping ::checkmodelfactors
overtoppingmessages ::getovertoppingparameter

Here is the caller graph for this function:


zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf
omkeervariantmodule ::checkifonberm

4.12.2.3 subroutine, public zfunctionsovertopping::profileinstructure (integer, intent(in) *nrCoordinates,* real(kind=wp), dimension(nrcoordinates), intent(in) *xcoordinates,* real(kind=wp), dimension(nrcoordinates), intent(in) *ycoordinates,* real(kind=wp), intent(in) *dikeHeight,* integer, intent(out) *nrCoordsAdjusted,* real(kind=wp), dimension(:), pointer *xCoordsAdjusted,* real(kind=wp), dimension(:), pointer *zCoordsAdjusted,* logical, intent(out) *succes,* character(len=∗), intent(out) *errorMessage*)

Subroutine to fill the profile in a structure and call the adjustment function of the profile due to a desired dike height.

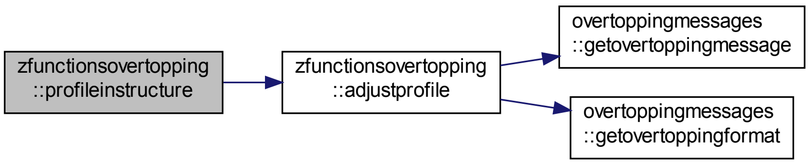
Parameters

`in`	*nrcoordinates*	number of coordinates of the profile
`in`	*xcoordinates*	vector with x-coordinates of the profile
`in`	*ycoordinates*	vector with y-coordinates of the profile
`in`	*dikeheight*	dike height
`out`	*nrcoordsad-justed*	number of coordinates in the adjusted profile

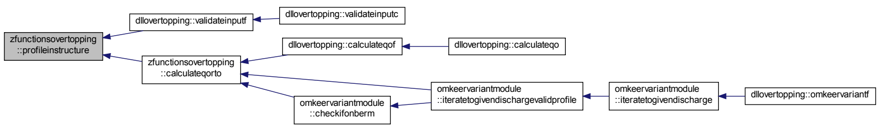
	xcoordsadjusted	vector with x-coordinates of the adjusted profile
	zcoordsadjusted	vector with y-coordinates of the adjusted profile
`out`	*succes*	flag for succes
`out`	*errormessage*	error message

Definition at line 104 of file zFunctionsOvertopping.f90.

Here is the call graph for this function:


zfunctionsovertopping ::profileinstructure
zfunctionsovertopping ::adjustprofile
overtoppingmessages ::getovertoppingmessage
overtoppingmessages ::getovertoppingformat

Here is the caller graph for this function:


zfunctionsovertopping ::profileinstructure
dllovertopping::validateinputf
dllovertopping::validateinputc
zfunctionsovertopping ::calculateqorto
dllovertopping::calculateqof
dllovertopping::calculateqo
omkeervariantmodule ::checkifonberm
omkeervariantmodule ::iteratetogivendischargevalidprofile
omkeervariantmodule ::iteratetogivendischarge
dllovertopping::omkeervariantf

4.12.2.4 real (kind=wp) function, public zfunctionsovertopping::zfunclogratios (real (kind=wp), intent(in) *qo*, real (kind=wp), intent(in) *qc*, real (kind=wp), intent(in) *mqo*, real (kind=wp), intent(in) *mqc*, logical, intent(out) *success*, character(len=∗), intent(out) *errorMessage*)

Routine to compute the limit state value by using the logs of the overtopping discharges (computed and desired)

Parameters

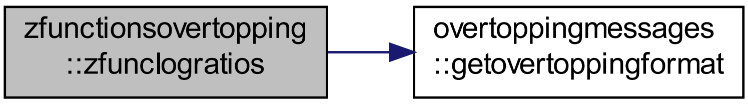
`in`	*qo*	computed overtopping discharge
`in`	*qc*	Critical overtopping discharge
`in`	*mqo*	Model factor computed overtopping discharge
`in`	*mqc*	Model factor Critical overtopping discharge
`out`	*success*	Flag for succes
`out`	*errormessage*	error message, only set if not successful

Returns

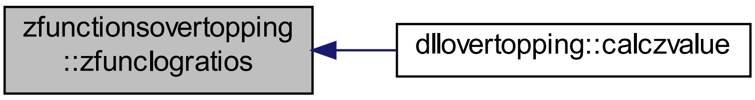
Value z-function

Definition at line 233 of file zFunctionsOvertopping.f90.

Here is the call graph for this function:


zfunctionsovertopping ::zfunclogratios
overtoppingmessages ::getovertoppingformat

Here is the caller graph for this function:

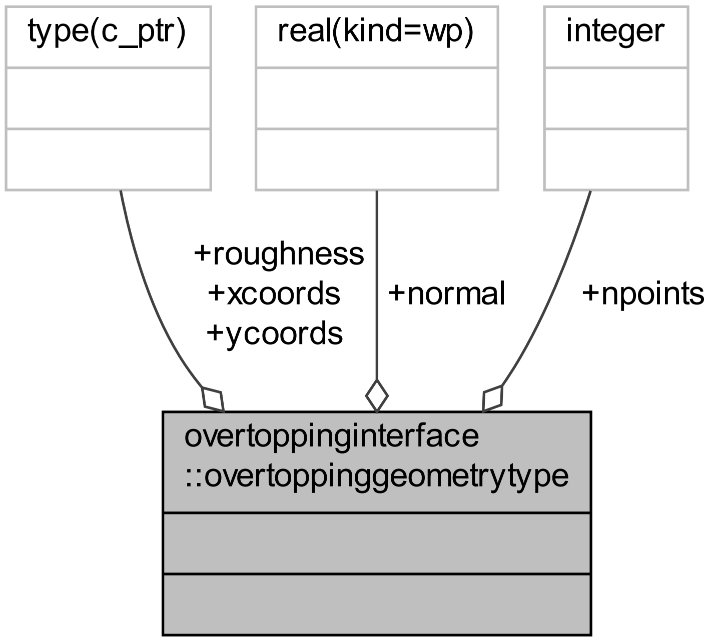

zfunctionsovertopping ::zfunclogratios
dllovertopping::calczvalue

Chapter 5

Data Type Documentation

5.1 overtoppinginterface::overtoppinggeometrytype Type Reference

Collaboration diagram for overtoppinginterface::overtoppinggeometrytype:



Public Attributes

- real(kind=wp) normal
- integer npoints
- type(c_ptr) xcoords
- type(c_ptr) ycoords
- type(c_ptr) roughness

5.1.1 Detailed Description

Definition at line 46 of file overtoppingInterface.f90.

5.1.2 Member Data Documentation

5.1.2.1 real(kind=wp) overtoppinginterface::overtoppinggeometrytype::normal

Definition at line 47 of file overtoppingInterface.f90.

5.1.2.2 integer overtoppinginterface::overtoppinggeometrytype::npoints

Definition at line 48 of file overtoppingInterface.f90.

5.1.2.3 type(c_ptr) overtoppinginterface::overtoppinggeometrytype::roughness

Definition at line 51 of file overtoppingInterface.f90.

5.1.2.4 type(c_ptr) overtoppinginterface::overtoppinggeometrytype::xcoords

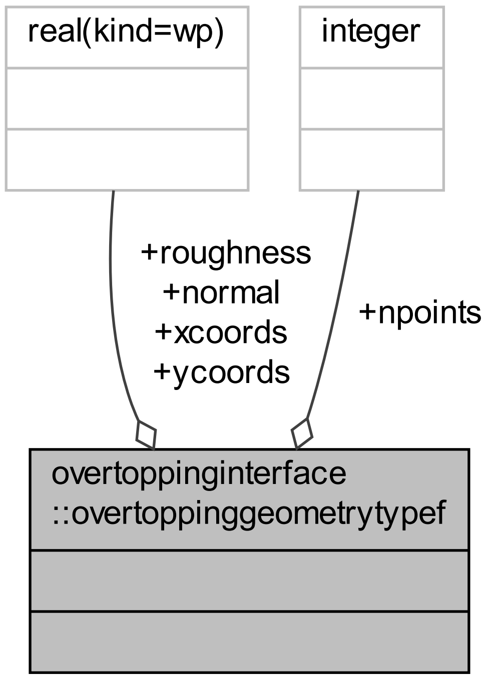
Definition at line 49 of file overtoppingInterface.f90.

5.1.2.5 type(c_ptr) overtoppinginterface::overtoppinggeometrytype::ycoords

Definition at line 50 of file overtoppingInterface.f90.

5.2 overtoppinginterface::overtoppinggeometrytypef Type Reference

Collaboration diagram for overtoppinginterface::overtoppinggeometrytypef:



Public Attributes

- real(kind=wp) normal
- integer npoints
- real(kind=wp), dimension(:), pointer xcoords
- real(kind=wp), dimension(:), pointer ycoords
- real(kind=wp), dimension(:), pointer roughness

5.2.1 Detailed Description

Definition at line 54 of file overtoppingInterface.f90.

5.2.2 Member Data Documentation

5.2.2.1 real(kind=wp) overtoppinginterface::overtoppinggeometrytypef::normal

Definition at line 55 of file overtoppingInterface.f90.

5.2.2.2 integer overtoppinginterface::overtoppinggeometrytypef::npoints

Definition at line 56 of file overtoppingInterface.f90.

5.2.2.3 real(kind=wp), dimension(:), pointer overtoppinginterface::overtoppinggeometrytypef::roughness

Definition at line 59 of file overtoppingInterface.f90.

5.2.2.4 real(kind=wp), dimension(:), pointer overtoppinginterface::overtoppinggeometrytypef::xcoords

Definition at line 57 of file overtoppingInterface.f90.

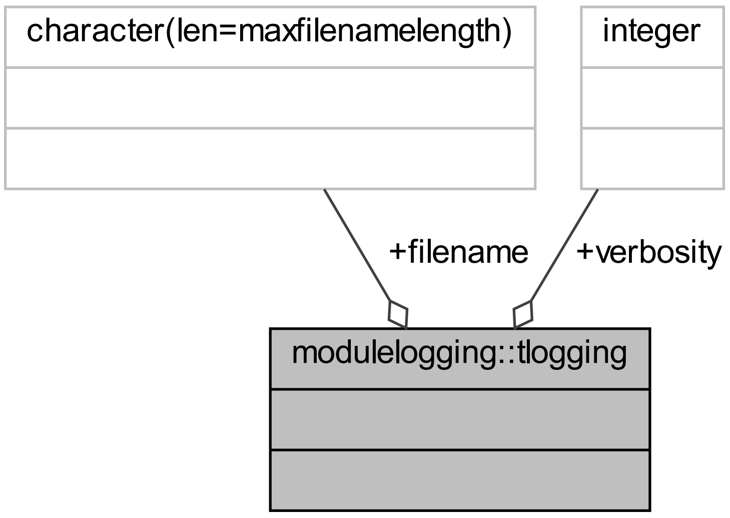
5.2.2.5 real(kind=wp), dimension(:), pointer overtoppinginterface::overtoppinggeometrytypef::ycoords

Definition at line 58 of file overtoppingInterface.f90.

5.3 modulelogging::tlogging Type Reference

TLogging: structure for steering the logging.

Collaboration diagram for modulelogging::tlogging:



Public Attributes

- integer verbosity = verboseNone

  level of verbosity: one of verboseNone, verboseBasic, verboseDetailed, verboseDebugging
- character(len=maxfilenamelength) filename = ' '

  filename of logging

5.3.1 Detailed Description

TLogging: structure for steering the logging.

Definition at line 38 of file ModuleLogging.f90.

5.3.2 Member Data Documentation

5.3.2.1 character(len=maxfilenamelength) modulelogging::tlogging::filename = ' '

filename of logging

Definition at line 40 of file ModuleLogging.f90.

5.3.2.2 integer modulelogging::tlogging::verbosity = verboseNone

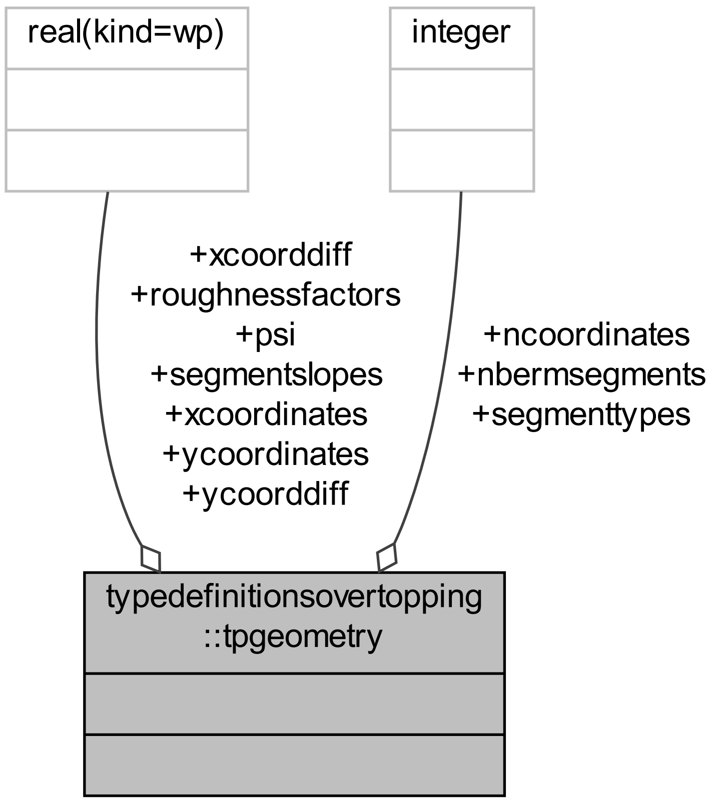
level of verbosity: one of verboseNone, verboseBasic, verboseDetailed, verboseDebugging

Definition at line 39 of file ModuleLogging.f90.

5.4 typedefinitionsovertopping::tpgeometry Type Reference

tpGeometry: structure with geometry data

Collaboration diagram for typedefinitionsovertopping::tpgeometry:



Public Attributes

- real(kind=wp) psi

dike normal (degrees)

- integer ncoordinates

  number of coordinates cross section
- real(kind=wp), dimension(:), pointer xcoordinates => null()

  vector with x-coordinates cross section (m)
- real(kind=wp), dimension(:), pointer ycoordinates => null()

  vector with y-coordinates cross section (m+NAP)
- real(kind=wp), dimension(:), pointer roughnessfactors => null()

  vector with roughness factors cross section
- real(kind=wp), dimension(:), pointer xcoorddiff => null()

  vector with differences in x-coordinates (m)
- real(kind=wp), dimension(:), pointer ycoorddiff => null()

  vector with differences in y-coordinates (m)
- real(kind=wp), dimension(:), pointer segmentslopes => null()

  vector with slopes dike segments
- integer, dimension(:), pointer segmenttypes

  vector with segment types (1=slope,2=berm,3=other)
- integer nbermsegments

  number of berm segments

5.4.1 Detailed Description

tpGeometry: structure with geometry data

Definition at line 42 of file typeDefinitionsOvertopping.f90.

5.4.2 Member Data Documentation

5.4.2.1 integer typedefinitionsovertopping::tpgeometry::nbermsegments

number of berm segments

Definition at line 52 of file typeDefinitionsOvertopping.f90.

5.4.2.2 integer typedefinitionsovertopping::tpgeometry::ncoordinates

number of coordinates cross section

Definition at line 44 of file typeDefinitionsOvertopping.f90.

5.4.2.3 real(kind=wp) typedefinitionsovertopping::tpgeometry::psi

dike normal (degrees)

Definition at line 43 of file typeDefinitionsOvertopping.f90.

5.4.2.4 real(kind=wp), dimension(:), pointer typedefinitionsovertopping::tpgeometry::roughnessfactors => null()

vector with roughness factors cross section

Definition at line 47 of file typeDefinitionsOvertopping.f90.

5.4.2.5 real(kind=wp), dimension(:), pointer typedefinitionsovertopping::tpgeometry::segmentslopes => null()

vector with slopes dike segments

Definition at line 50 of file typeDefinitionsOvertopping.f90.

5.4.2.6 integer, dimension(:), pointer typedefinitionsovertopping::tpgeometry::segmenttypes

vector with segment types (1=slope,2=berm,3=other)

Definition at line 51 of file typeDefinitionsOvertopping.f90.

5.4.2.7 real(kind=wp), dimension(:), pointer typedefinitionsovertopping::tpgeometry::xcoorddiff => null()

vector with differences in x-coordinates (m)

Definition at line 48 of file typeDefinitionsOvertopping.f90.

5.4.2.8 real(kind=wp), dimension(:), pointer typedefinitionsovertopping::tpgeometry::xcoordinates => null()

vector with x-coordinates cross section (m)

Definition at line 45 of file typeDefinitionsOvertopping.f90.

5.4.2.9 real(kind=wp), dimension(:), pointer typedefinitionsovertopping::tpgeometry::ycoorddiff => null()

vector with differences in y-coordinates (m)

Definition at line 49 of file typeDefinitionsOvertopping.f90.

5.4.2.10 real(kind=wp), dimension(:), pointer typedefinitionsovertopping::tpgeometry::ycoordinates => null()

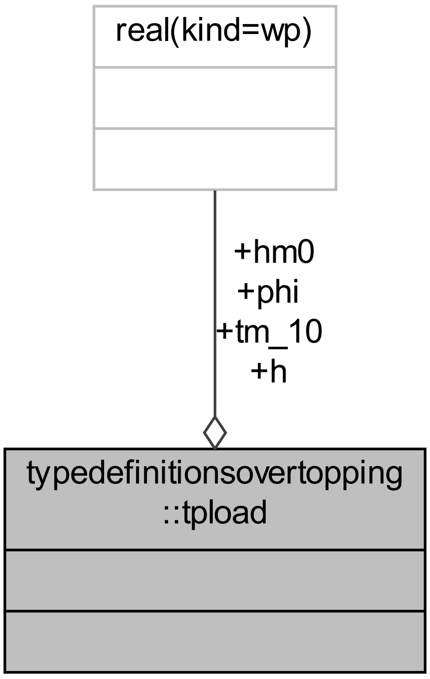
vector with y-coordinates cross section (m+NAP)

Definition at line 46 of file typeDefinitionsOvertopping.f90.

5.5 typedefinitionsovertopping::tpload Type Reference

tpLoad: structure with load parameters

Collaboration diagram for typedefinitionsovertopping::tpload:



Public Attributes

- real(kind=wp) h

    local water level (m+NAP)
- real(kind=wp) hm0

    significant wave height (m)
- real(kind=wp) tm_10

    spectral wave period (s)
- real(kind=wp) phi

    wave direction (degrees)

5.5.1 Detailed Description

tpLoad: structure with load parameters

Definition at line 56 of file typeDefinitionsOvertopping.f90.

5.5.2 Member Data Documentation

5.5.2.1 real(kind=wp) typedefinitionsovertopping::tpload::h

local water level (m+NAP)

Definition at line 57 of file typeDefinitionsOvertopping.f90.

5.5.2.2 real(kind=wp) typedefinitionsovertopping::tpload::hm0

significant wave height (m)

Definition at line 58 of file typeDefinitionsOvertopping.f90.

5.5.2.3 real(kind=wp) typedefinitionsovertopping::tpload::phi

wave direction (degrees)

Definition at line 60 of file typeDefinitionsOvertopping.f90.

5.5.2.4 real(kind=wp) typedefinitionsovertopping::tpload::tm_10

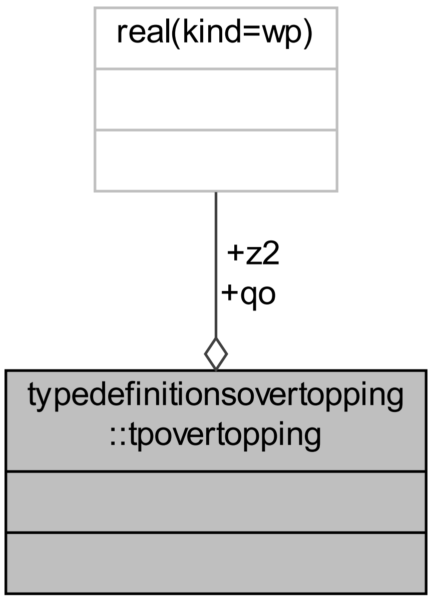
spectral wave period (s)

Definition at line 59 of file typeDefinitionsOvertopping.f90.

5.6 typedefinitionsovertopping::tpovertopping Type Reference

tpOvertopping: structure with overtopping results

Collaboration diagram for typedefinitionsovertopping::tpovertopping:



Public Attributes

- real(kind=wp) z2

    2% wave run-up (m)
- real(kind=wp) qo

wave overtopping discharge (m3/m per s)

5.6.1 Detailed Description

tpOvertopping: structure with overtopping results

Definition at line 80 of file typeDefinitionsOvertopping.f90.

5.6.2 Member Data Documentation

5.6.2.1 real(kind=wp) typedefinitionsovertopping::tpovertopping::qo

wave overtopping discharge (m3/m per s)

Definition at line 82 of file typeDefinitionsOvertopping.f90.

5.6.2.2 real(kind=wp) typedefinitionsovertopping::tpovertopping::z2

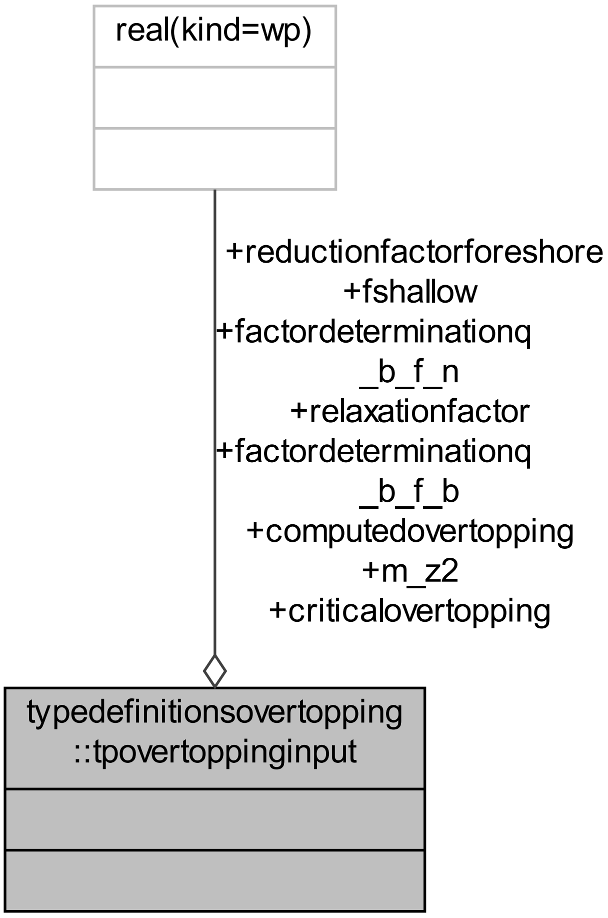
2% wave run-up (m)

Definition at line 81 of file typeDefinitionsOvertopping.f90.

5.7 typedefinitionsovertopping::tpovertoppinginput Type Reference

OvertoppingModelFactors: C-structure with model factors.

Collaboration diagram for typedefinitionsovertopping::tpovertoppinginput:


real(kind=wp)
+reductionfactorforeshore
+fshallow
+factordeterminationq_b_f_n
+relaxationfactor
+factordeterminationq_b_f_b
+computedovertopping
+m_z2
+criticalovertopping
typedefinitionsovertopping::tpovertoppinginput

Public Attributes

- real(kind=wp) factordeterminationq_b_f_n

  model factor for non-breaking waves
- real(kind=wp) factordeterminationq_b_f_b

  model factor for breaking waves
- real(kind=wp) m_z2

  model factor describing the uncertainty of 2% runup height
- real(kind=wp) fshallow

  model factor for shallow waves
- real(kind=wp) computedovertopping

  model factor computed overtopping
- real(kind=wp) criticalovertopping

  model factor critical overtopping
- real(kind=wp) relaxationfactor

  relaxation factor iteration procedure wave runup

- real(kind=wp) reductionfactorforeshore = 0.5_wp

  reduction factor foreshore

5.7.1 Detailed Description

OvertoppingModelFactors: C-structure with model factors.

Definition at line 68 of file typeDefinitionsOvertopping.f90.

5.7.2 Member Data Documentation

5.7.2.1 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::computedovertopping

model factor computed overtopping

Definition at line 73 of file typeDefinitionsOvertopping.f90.

5.7.2.2 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::criticalovertopping

model factor critical overtopping

Definition at line 74 of file typeDefinitionsOvertopping.f90.

5.7.2.3 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::factordeterminationq_b_f_b

model factor for breaking waves

Definition at line 70 of file typeDefinitionsOvertopping.f90.

5.7.2.4 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::factordeterminationq_b_f_n

model factor for non-breaking waves

Definition at line 69 of file typeDefinitionsOvertopping.f90.

5.7.2.5 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::fshallow

model factor for shallow waves

Definition at line 72 of file typeDefinitionsOvertopping.f90.

5.7.2.6 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::m_z2

model factor describing the uncertainty of 2% runup height

Definition at line 71 of file typeDefinitionsOvertopping.f90.

5.7.2.7 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::reductionfactorforeshore = 0.5_wp

reduction factor foreshore

Definition at line 76 of file typeDefinitionsOvertopping.f90.

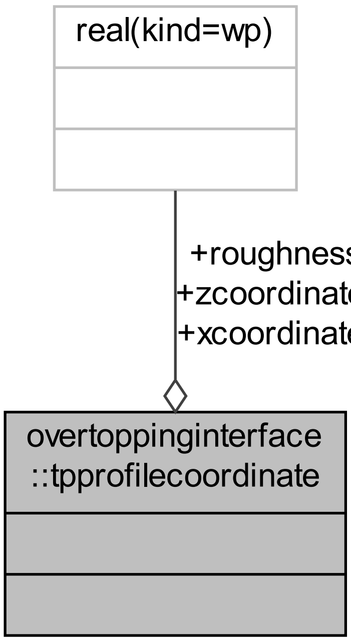
5.7.2.8 real(kind=wp) typedefinitionsovertopping::tpovertoppinginput::relaxationfactor

relaxation factor iteration procedure wave runup

Definition at line 75 of file typeDefinitionsOvertopping.f90.

5.8 overtoppinginterface::tpprofilecoordinate Type Reference

Collaboration diagram for overtoppinginterface::tpprofilecoordinate:



Public Attributes

- real(kind=wp) xcoordinate

    X-coordinate foreland profile.
- real(kind=wp) zcoordinate

    Z-coordinate foreland profile.
- real(kind=wp) roughness

    Roughness of the area between two points.

5.8.1 Detailed Description

Definition at line 40 of file overtoppingInterface.f90.

5.8.2 Member Data Documentation

5.8.2.1 real(kind=wp) overtoppinginterface::tpprofilecoordinate::roughness

Roughness of the area between two points.

Definition at line 43 of file overtoppingInterface.f90.

5.8.2.2 real(kind=wp) overtoppinginterface::tpprofilecoordinate::xcoordinate

X-coordinate foreland profile.

Definition at line 41 of file overtoppingInterface.f90.

5.8.2.3 real(kind=wp) overtoppinginterface::tpprofilecoordinate::zcoordinate

Z-coordinate foreland profile.

Definition at line 42 of file overtoppingInterface.f90.

Chapter 6

File Documentation

6.1 dllOvertopping.f90 File Reference

Main entry for the dll DikesOvertopping FUNCTIONS/SUBROUTINES exported from dllOvertopping.dll:

Modules

- module dllovertopping

  Main entry for the dll DikesOvertopping.

Functions/Subroutines

- subroutine, public dllovertopping::calculateqo (load, geometryInput, dikeHeight, modelFactors, overtopping, success, errorText, verbosity, logFile)

  Subroutine that calculates the discharge needed for the Z-function DikesOvertopping Wrapper for calculateQoF↩ : convert C-like input structures to Fortran input structures.
- subroutine, public dllovertopping::calculateqof (load, geometryF, dikeHeight, modelFactors, overtopping, success, errorText, logging)

  Subroutine that calculates the discharge needed for the Z-function DikesOvertopping.
- subroutine, public dllovertopping::calczvalue (criticalOvertoppingRate, modelFactors, Qo, z, success, error↩ Message)

  Subroutine that calculates the Z-function DikesOvertopping based on the discharge calculated with calculateQoF.
- subroutine, public dllovertopping::validateinputc (geometryInput, dikeHeight, modelFactors, success, error↩ Text)

  Subroutine that validates the geometry Wrapper for ValidateInputFold: convert C-like input structures to Fortran input structures.
- subroutine, public dllovertopping::validateinputf (geometryF, dikeHeight, modelFactors, errorStruct)

  Subroutine that validates the geometry.
- subroutine, public dllovertopping::omkeervariantf (load, geometryF, givenDischarge, dikeHeight, model↩ Factors, overtopping, success, errorText, logging)

  Subroutine with omkeerVariant.
- subroutine, public dllovertopping::setlanguage (lang)

  Subroutine that sets the language for error and validation messages.
- subroutine, public dllovertopping::getlanguage (lang)

  Subroutine that gets the language for error and validation messages.
- subroutine, public dllovertopping::versionnumber (version)

  Subroutine that delivers the version number.
- type(overtoppinggeometrytypef) function dllovertopping::geometry_c_f (geometryInput)

  Private subroutine that converts geometry from c-pointer to fortran struct.

6.1.1 Detailed Description

Main entry for the dll DikesOvertopping FUNCTIONS/SUBROUTINES exported from dllOvertopping.dll:

- calcZValue
- calculateQo
- calculateQoF
- ValidateInputC
- ValidateInputF
- omkeerVariantF
- SetLanguage
- GetLanguage
- versionNumber

6.2 factorModuleOvertopping.f90 File Reference

This file contains a module with functions for the slope angle and influence factors.

Modules

- module factormoduleovertopping

  functions for the slope angle and influence factors

Functions/Subroutines

- subroutine, public factormoduleovertopping::calculatetanalpha (h, Hm0, z2, geometry, tanAlpha, succes, errorMessage)

  calculateTanAlpha representative slope angle
- subroutine, public factormoduleovertopping::calculategammabeta (Hm0, Tm_10, beta, gammaBeta_z, gammaBeta_o)

  calculateGammaBeta influence factor angle of wave attack
- subroutine, public factormoduleovertopping::calculategammaf (h, ksi0, ksi0Limit, gammaB, z2, geometry, gammaF, succes, errorMessage)

  calculateGammaF influence factor roughness
- subroutine, public factormoduleovertopping::calculategammab (h, Hm0, z2, geometry, gammaB, succes, errorMessage)

  calculateGammaB influence factor berms

6.2.1 Detailed Description

This file contains a module with functions for the slope angle and influence factors.

6.3 formulaModuleOvertopping.f90 File Reference

This file contains a module with the core computations for Dikes Overtopping.

Modules

- module formulamoduleovertopping

    the core computations for Dikes Overtopping

Functions/Subroutines

- subroutine, public formulamoduleovertopping::calculatewaverunup (Hm0, ksi0, ksi0Limit, gammaB, gammaF, gammaBeta, modelFactors, z2, succes, errorMessage)

    calculateWaveRunup: calculate wave runup
- subroutine, public formulamoduleovertopping::calculatewaveovertoppingdischarge (h, Hm0, tanAlpha, gammaB, gammaF, gammaBeta, ksi0, hCrest, modelFactors, Qo, succes, errorMessage)

    calculateWaveOvertoppingDischarge: calculate the wave overtopping discharge
- subroutine, public formulamoduleovertopping::calculatewavelength (Tm_10, L0)

    calculateWaveLength: calculate the wave length
- subroutine, public formulamoduleovertopping::calculatewavesteepness (Hm0, Tm_10, s0, succes, error←
Message)

    calculateWaveSteepness: calculate the wave steepness
- subroutine, public formulamoduleovertopping::calculatebreakerparameter (tanAlpha, s0, ksi0, succes, error←
Message)

    calculateBreakerParameter: calculate the breaker parameter
- subroutine, public formulamoduleovertopping::calculateanglewaveattack (phi, psi, beta)

    calculateAngleWaveAttack: calculate the angle of wave attack
- subroutine, public formulamoduleovertopping::calculatebreakerlimit (gammaB, ksi0Limit, succes, error←
Message)

    calculateBreakerLimit: calculate the breaker limit
- subroutine, public formulamoduleovertopping::adjustinfluencefactors (gammaB, gammaF, gammaBeta, gammaBetaType, ksi0, ksi0Limit, succes, errorMessage)

    adjustInfluenceFactors: adjust the influence factors
- subroutine formulamoduleovertopping::realrootscubicfunction (a, b, c, d, N, x, succes, errorMessage)

    realRootsCubicFunction: calculate the roots of a cubic function
- subroutine formulamoduleovertopping::rootsgeneralcubic (a, b, c, d, z, succes, errorMessage)

    rootsGeneralCubic: calculate the roots of a generic cubic function
- subroutine formulamoduleovertopping::rootsdepressedcubic (p, q, z)

    rootsDepressedCubic: calculate the roots of a depressed cubic function
- subroutine formulamoduleovertopping::cubicroots (z, roots)

    cubicRoots: calculate the roots of a cubic function
- logical function, public formulamoduleovertopping::isequalreal (x1, x2)

    isEqualReal: are two reals (almost) equal
- logical function, public formulamoduleovertopping::isequalzero (x)

    isEqualZero: is a real (almost) zero

6.3.1 Detailed Description

This file contains a module with the core computations for Dikes Overtopping.

6.4 geometryModuleOvertopping.f90 File Reference

This file contains a module with the core computations for Dikes Overtopping related to the geometry.

Modules

- module geometrymoduleovertopping

    core computations related to the geometry

Functions/Subroutines

- subroutine, public geometrymoduleovertopping::checkcrosssection (psi, nCoordinates, xCoordinates, y←↩ Coordinates, roughnessFactors, succes, errorMessage)

    checkCrossSection: check cross section
- subroutine, public geometrymoduleovertopping::initializegeometry (psi, nCoordinates, xCoordinates, y←↩ Coordinates, roughnessFactors, geometry, succes, errorMessage)

    initializeGeometry: initialize the geometry
- subroutine, public geometrymoduleovertopping::allocatevectorsgeometry (nCoordinates, geometry, succes, errorMessage)

    allocateVectorsGeometry: allocate the geometry vectors
- subroutine, public geometrymoduleovertopping::deallocategeometry (geometry)

    deallocateGeometry: deallocate the geometry vectors
- subroutine, public geometrymoduleovertopping::calculatesegmentslopes (geometry, succes, errorMessage)

    calculateSegmentSlopes: calculate the segment slopes
- subroutine, public geometrymoduleovertopping::determinesegmenttypes (geometry)

    determineSegmentTypes: determine the segment types
- subroutine, public geometrymoduleovertopping::copygeometry (geometry, geometryCopy, succes, error←↩ Message)

    copyGeometry: copy a geometry structure
- subroutine, public geometrymoduleovertopping::mergesequentialberms (geometry, geometryMergedBerms, succes, errorMessage)

    mergeSequentialBerms: merge sequential berms
- subroutine, public geometrymoduleovertopping::adjustnonhorizontalberms (geometry, geometryFlatBerms, succes, errorMessage)

    adjustNonHorizontalBerms: adjust non-horizontal berms
- subroutine, public geometrymoduleovertopping::removeberms (geometry, geometryNoBerms, succes, errorMessage)

    removeBerms: remove berms
- subroutine, public geometrymoduleovertopping::removedikesegments (geometry, index, geometryAdjusted, succes, errorMessage)

    removeDikeSegments: remove dike segments
- subroutine, public geometrymoduleovertopping::splitcrosssection (geometry, L0, NwideBerms, geometrysection←↩ B, geometrysectionF, succes, errorMessage)

    splitCrossSection: split a cross section
- subroutine, public geometrymoduleovertopping::calculatehorzlengths (geometry, yLower, yUpper, horz←↩ Lengths, succes, errorMessage)

    calculateHorzLengths: calculate horizontal lengths
- subroutine, public geometrymoduleovertopping::calculatehorzdistance (geometry, yLower, yUpper, dx, succes, errorMessage)

    calculateHorzDistance: calculate horizontal distance
- subroutine, public geometrymoduleovertopping::basicgeometrytest (geometryF, success, errorStruct)

    basicGeometryTest: test the input geometry (the adjusted geometry is checked elsewhere)

6.4.1 Detailed Description

This file contains a module with the core computations for Dikes Overtopping related to the geometry.

6.5 mainModuleOvertopping.f90 File Reference

This file contains a module with the core computations for Dikes Overtopping.

Modules

- module mainmoduleovertopping

  core computations for Dikes Overtopping

Functions/Subroutines

- subroutine, public mainmoduleovertopping::calculateovertopping (geometry, load, modelFactors, overtopping, succes, errorMessage)

  calculateOvertopping: calculate the overtopping
- subroutine, public mainmoduleovertopping::calculateovertoppingsection (geometry, h, Hm0, Tm_10, L0, gammaBeta_z, gammaBeta_o, modelFactors, overtopping, succes, errorMessage)

  calculateOvertoppingSection: calculate the overtopping for a section
- subroutine, public mainmoduleovertopping::calculatewaveovertopping (geometry, h, Hm0, Tm_10, z2, gammaBeta_o, modelFactors, Qo, succes, errorMessage)

  calculateWaveOvertopping: calculate wave overtopping
- subroutine mainmoduleovertopping::calculateovertoppingnegativefreeboard (load, geometry, overtopping, succes, errorMessage)

  calculateOvertoppingNegativeFreeboard: calculate overtopping in case of negative freeboard
- subroutine, public mainmoduleovertopping::interpolateresultssections (geometry, L0, NwideBerms, overtoppingB, overtoppingF, overtopping, succes, errorMessage)

  interpolateResultsSections: interpolate results for split cross sections
- subroutine, public mainmoduleovertopping::checkinputdata (geometry, load, modelFactors, succes, error↩ Message)

  checkInputdata: check the input data
- subroutine, public mainmoduleovertopping::checkmodelfactors (modelFactors, dimErrMessage, error↩ Messages, ierr)

  checkModelFactors: check the input data

6.5.1 Detailed Description

This file contains a module with the core computations for Dikes Overtopping.

6.6 ModuleLogging.f90 File Reference

Module for steering the extra logging.

Data Types

- type modulelogging::tlogging

  TLogging: structure for steering the logging.

Modules

- module modulelogging

  steering the extra logging

Variables

- integer, parameter modulelogging::maxfilenamelength = 256

  maximum length of filename
- type(tlogging) modulelogging::currentlogging

  copy of argument logging

6.6.1 Detailed Description

Module for steering the extra logging.

6.7 omkeerVariantModule.f90 File Reference

This file contains the omkeerVariant.

Modules

- module omkeervariantmodule

  Module for the 'omkeerVariant'.

Functions/Subroutines

- subroutine, public omkeervariantmodule::iteratetogivendischarge (load, geometryF, givenDischarge, dike←↩ Height, modelFactors, overtopping, success, errorText, logging)

  Subroutine with omkeerVariant.
- subroutine omkeervariantmodule::iteratetogivendischargevalidprofile (load, geometry, givenDischarge, dike←↩ Height, modelFactors, overtopping, success, errorText)

  Subroutine with iterateToGivenDischarge, with already checked profile.
- subroutine omkeervariantmodule::checkifonberm (geometry, load, modelfactors, overtopping, given←↩ Discharge, dikeHeight, iUp, iLow, dis1, dis2, minDikeHeight, maxDikeHeight, foundValue, success, errorText)

Variables

- real(kind=wp), parameter omkeervariantmodule::toldischarge = 1d-3
- real(kind=wp), parameter omkeervariantmodule::toldikeheight = 1d-3
- real(kind=wp), parameter omkeervariantmodule::minzberm = 0.1_wp
- logical, dimension(:), allocatable omkeervariantmodule::isberm
- logical, dimension(:), allocatable omkeervariantmodule::isvalidz
- real(kind=wp), dimension(:), allocatable omkeervariantmodule::dischargeprofile
- real(kind=wp), dimension(:), allocatable omkeervariantmodule::zprofile

6.7.1 Detailed Description

This file contains the omkeerVariant.

6.8 overtoppingInterface.f90 File Reference

This file contains the parameters and types (structs) as part of the interface to and from dllOvertopping.

Data Types

- type overtoppinginterface::tpprofilecoordinate
- type overtoppinginterface::overtoppinggeometrytype
- type overtoppinginterface::overtoppinggeometrytypef

Modules

- module overtoppinginterface

  Module for the interface of dllOvertopping.

Variables

- integer, parameter, public overtoppinginterface::varmodelfactorcriticalovertopping = 8

  Model factor critical overtopping.

6.8.1 Detailed Description

This file contains the parameters and types (structs) as part of the interface to and from dllOvertopping.

6.9 OvertoppingMessages.f90 File Reference

This file contains the messages in the overtopping dll, in Dutch or English.

Modules

- module overtoppingmessages

  Module for the messages in the overtopping dll, in Dutch or English.

Functions/Subroutines

- subroutine overtoppingmessages::setlanguage (lang)

  IDs for the strings in this module:
- subroutine overtoppingmessages::getlanguage (lang)

  Subroutine that gets the language for error and validation messages.
- character(len=maxmsg) function overtoppingmessages::getovertoppingmessage (ID)

  Subroutine that returns a message with the corresponding ID in the current language.
- character(len=maxmsg) function overtoppingmessages::getovertoppingformat (ID)

  Subroutine that returns a Fortran format string with the corresponding ID in the current language.
- character(len=maxpar) function overtoppingmessages::getovertoppingparameter (ID)

  Subroutine that returns the name of an input parameter with the corresponding ID in the current language.

Variables

- integer, parameter, private overtoppingmessages::maxmsg = 128
- integer, parameter, private overtoppingmessages::maxpar =32
- character(len=2), private overtoppingmessages::language = 'NL'

  default : Dutch

6.9.1 Detailed Description

This file contains the messages in the overtopping dll, in Dutch or English.

6.10 typeDefinitionsOvertopping.f90 File Reference

This file contains a module with the type definitions for Dikes Overtopping.

Data Types

- type typedefinitionsovertopping::tpgeometry

  tpGeometry: structure with geometry data
- type typedefinitionsovertopping::tpload

  tpLoad: structure with load parameters
- type typedefinitionsovertopping::tpovertoppinginput

  OvertoppingModelFactors: C-structure with model factors.
- type typedefinitionsovertopping::tpovertopping

  tpOvertopping: structure with overtopping results

Modules

- module typedefinitionsovertopping

  type definitions for Dikes Overtopping

Variables

- real(kind=wp), parameter typedefinitionsovertopping::frunup1 = 1.65_wp
- real(kind=wp), parameter typedefinitionsovertopping::frunup2 = 4.00_wp
- real(kind=wp), parameter typedefinitionsovertopping::frunup3 = 1.50_wp
- real(kind=wp), parameter typedefinitionsovertopping::xdiff_min = 2.0d-2

  minimal value distance between x-coordinates (m)
- real(kind=wp), parameter typedefinitionsovertopping::margindiff = 1.0d-14

  margin for minimal distance (m)
- real(kind=wp), parameter typedefinitionsovertopping::berm_min = 0.0d0

  minimal value gradient berm segment
- real(kind=wp), parameter typedefinitionsovertopping::berm_max = 1.0d0/15

  maximal value gradient berm segment
- real(kind=wp), parameter typedefinitionsovertopping::slope_min = 1.0d0/8

  minimal value gradient slope segment
- real(kind=wp), parameter typedefinitionsovertopping::slope_max = 1.0d0

  maximal value gradient slope segment
- real(kind=wp), parameter typedefinitionsovertopping::margingrad = 0.0025d0

  margin for minimal and maximal gradients
- real(kind=wp), parameter typedefinitionsovertopping::rfactor_min = 0.5d0

  minimal value roughness factor dike segments
- real(kind=wp), parameter typedefinitionsovertopping::rfactor_max = 1.0d0

  maximal value roughness factor dike segments
- real(kind=wp), parameter typedefinitionsovertopping::mz2_min = 0.0d0

  minimal value model factor of 2% runup height

- real(kind=wp), parameter typedefinitionsovertopping::mz2_max = huge(mz2_max)

  maximal value model factor of 2% runup height
- real(kind=wp), parameter typedefinitionsovertopping::fb_min = 0.0d0

  minimal value model factor for breaking waves
- real(kind=wp), parameter typedefinitionsovertopping::fb_max = huge(fB_max)

  maximal value model factor for breaking waves
- real(kind=wp), parameter typedefinitionsovertopping::fn_min = 0.0d0

  minimal value model factor for non-breaking waves
- real(kind=wp), parameter typedefinitionsovertopping::fn_max = huge(fN_max)

  maximal value model factor for non-breaking waves
- real(kind=wp), parameter typedefinitionsovertopping::fs_min = 0.0d0

  minimal value model factor for shallow waves
- real(kind=wp), parameter typedefinitionsovertopping::fs_max = huge(fS_max)

  maximal value model factor for shallow waves
- real(kind=wp), parameter typedefinitionsovertopping::foreshore_min = 0.3d0

  minimal value reduction factor foreshore
- real(kind=wp), parameter typedefinitionsovertopping::foreshore_max = 1.0d0

  maximal value reduction factor foreshore
- integer, parameter typedefinitionsovertopping::z2_iter_max1 = 49

  maximal number of iterations for calculation z2 part 1
- integer, parameter typedefinitionsovertopping::z2_iter_max2 = 70

  maximal number of iterations for calculation z2 part 1 & 2
- real(kind=wp), parameter typedefinitionsovertopping::z2_margin = 0.001d0

  margin for convergence criterium calculation z2

6.10.1 Detailed Description

This file contains a module with the type definitions for Dikes Overtopping.

6.11 waveRunup.f90 File Reference

This file contains a module with the iteration procedure for 2% wave runup.

Modules

- module waverunup

  Iteration procedure for 2% wave runup.

Functions/Subroutines

- subroutine, public waverunup::iterationwaverunup (geometry, h, Hm0, Tm_10, gammaBeta_z, modelFactors, z2, succes, errorMessage)

  iterationWaveRunup: iteration for the wave runup
- real(kind=wp) function waverunup::innercalculation (geometry, h, Hm0, gammaBeta_z, modelFactors, z2, s0, geometryFlatBerms, succes, errorMessage)

  innerCalculation: inner calculation for the wave runup
- real(kind=wp) function waverunup::determinestartingvalue (i, relaxationFactor, z2_start, z2_end, Hm0)

  determineStartingValue: helper function to find a start value for z2
- integer function waverunup::findsmallestresidu (z2_start, z2_end, n)

findSmallestResidu: helper function to find the smallest residu

- subroutine waverunup::convergedwithresidu (z2_start, z2_end)

    convergedWithResidu: helper function to handle convergence with a higher residu than expected if logging is enabled, it writes a small message to the logfile

6.11.1 Detailed Description

This file contains a module with the iteration procedure for 2% wave runup.

6.12 zFunctionsOvertopping.f90 File Reference

This file contains the limit state functions for wave overtopping within VTV.

Modules

- module zfunctionsovertopping

    Module for the Limit State Functions (Z-functions) for wave overtopping.

Functions/Subroutines

- subroutine, public zfunctionsovertopping::calculateqorto (dikeHeight, modelFactors, overtopping, load, geometry, succes, errorMessage)

    Subroutine to calculate the overtopping discharge with the Overtopping dll.
- subroutine, public zfunctionsovertopping::profileinstructure (nrCoordinates, xcoordinates, ycoordinates, dikeHeight, nrCoordsAdjusted, xCoordsAdjusted, zCoordsAdjusted, succes, errorMessage)

    Subroutine to fill the profile in a structure and call the adjustment function of the profile due to a desired dike height.
- subroutine zfunctionsovertopping::adjustprofile (nrCoordinates, coordinates, dikeHeight, nrCoordsAdjusted, xCoordsAdjusted, zCoordsAdjusted, succes, errorMessage)

    Subroutine adjust the profile due to a desired dike height.
- real(kind=wp) function, public zfunctionsovertopping::zfunclogratios (qo, qc, mqo, mqc, success, error↩ Message)

    Routine to compute the limit state value by using the logs of the overtopping discharges (computed and desired)

6.12.1 Detailed Description

This file contains the limit state functions for wave overtopping within VTV.

Index