

javaSTGDH Reference Manual

Generated by Doxygen 1.4.6

Mon Jul 9 10:35:40 2007

Contents

1	S-TGDH: Secure enhanced group management protocol in ad hoc networks	1
1.1	Introduction	1
1.2	Utilizations	2
2	javaSTGDH Namespace Index	3
2.1	javaSTGDH Namespace List	3
3	javaSTGDH Hierarchical Index	5
3.1	javaSTGDH Class Hierarchy	5
4	javaSTGDH Class Index	7
4.1	javaSTGDH Class List	7
5	javaSTGDH File Index	9
5.1	javaSTGDH File List	9
6	javaSTGDH Namespace Documentation	11
6.1	console Namespace Reference	11
6.2	lang Namespace Reference	12
6.3	manager Namespace Reference	13
6.4	manager::messages Namespace Reference	14
6.5	node Namespace Reference	15
6.6	node::buffers Namespace Reference	16
6.7	node::messages Namespace Reference	17
6.8	tools Namespace Reference	18
6.9	ui Namespace Reference	19
7	javaSTGDH Class Documentation	21
7.1	ui::About Class Reference	21
7.2	console::Aleatory Class Reference	23

7.3	ui::CellStyle Class Reference	25
7.4	node::buffers::Challenge2Response Class Reference	28
7.5	node::buffers::ChallengeExpected Class Reference	30
7.6	ui::Constants Interface Reference	32
7.7	ui::CreateGroup Class Reference	35
7.8	tools::Crypto Class Reference	37
7.9	node::DelayException Class Reference	41
7.10	node::GroupNode Class Reference	42
7.11	node::IGroupNode Interface Reference	47
7.12	node::IInitNode Interface Reference	49
7.13	manager::ILauncher Interface Reference	50
7.14	node::InitNode Class Reference	51
7.15	node::INode Interface Reference	55
7.16	ui::Interface Class Reference	56
7.17	tools::JavaTools Class Reference	58
7.18	lang::Language Class Reference	61
7.19	lang::LanguageException Class Reference	63
7.20	console::Launcher Class Reference	65
7.21	manager::Launcher Class Reference	67
7.22	ui::Launcher Class Reference	69
7.23	manager::Log Class Reference	70
7.24	ui::Manager Class Reference	75
7.25	manager::messages::Message Class Reference	78
7.26	manager::messages::MulticastMessage Class Reference	81
7.27	node::Node Class Reference	83
7.28	node::NodeBehavior Class Reference	90
7.29	node::NodeConstants Interface Reference	92
7.30	ui::NodeFrame Class Reference	94
7.31	node::messages::NodeMessage Class Reference	95
7.32	manager::Supervisor Class Reference	97
7.33	manager::SupervisorMessage Interface Reference	98
7.34	tools::Tree Class Reference	102
7.35	tools::TreeNode Class Reference	107
7.36	ui::XGridBag Class Reference	113
8	javaSTGDH File Documentation	117
8.1	workspace/S-TGDH_god/console/Aleatory.java File Reference	117

8.2	workspace/S-TGDH_god/console/Launcher.java File Reference	118
8.3	workspace/S-TGDH_god/manager/Launcher.java File Reference	119
8.4	workspace/S-TGDH_god/ui/Launcher.java File Reference	120
8.5	workspace/S-TGDH_god/lang/Language.java File Reference	121
8.6	workspace/S-TGDH_god/lang/LanguageException.java File Reference	122
8.7	workspace/S-TGDH_god/manager/ILauncher.java File Reference	123
8.8	workspace/S-TGDH_god/manager/Log.java File Reference	124
8.9	workspace/S-TGDH_god/manager/messages/Message.java File Reference	125
8.10	workspace/S-TGDH_god/manager/messages/MulticastMessage.java File Reference	126
8.11	workspace/S-TGDH_god/manager/Supervisor.java File Reference	127
8.12	workspace/S-TGDH_god/manager/SupervisorMessage.java File Reference	128
8.13	workspace/S-TGDH_god/namespace.doxydoc File Reference	129
8.14	workspace/S-TGDH_god/node/buffers/Challenge2Response.java File Reference	130
8.15	workspace/S-TGDH_god/node/buffers/ChallengeExpected.java File Reference	131
8.16	workspace/S-TGDH_god/node/DelayException.java File Reference	132
8.17	workspace/S-TGDH_god/node/GroupNode.java File Reference	133
8.18	workspace/S-TGDH_god/node/IGroupNode.java File Reference	134
8.19	workspace/S-TGDH_god/node/IInitNode.java File Reference	135
8.20	workspace/S-TGDH_god/node/InitNode.java File Reference	136
8.21	workspace/S-TGDH_god/node/INode.java File Reference	137
8.22	workspace/S-TGDH_god/node/messages/NodeMessage.java File Reference	138
8.23	workspace/S-TGDH_god/node/Node.java File Reference	139
8.24	workspace/S-TGDH_god/node/NodeBehavior.java File Reference	140
8.25	workspace/S-TGDH_god/node/NodeConstants.java File Reference	141
8.26	workspace/S-TGDH_god/tools/Crypto.java File Reference	142
8.27	workspace/S-TGDH_god/tools/JavaTools.java File Reference	143
8.28	workspace/S-TGDH_god/tools/Tree.java File Reference	144
8.29	workspace/S-TGDH_god/tools/TreeNode.java File Reference	145
8.30	workspace/S-TGDH_god/ui/About.java File Reference	146
8.31	workspace/S-TGDH_god/ui/CellStyle.java File Reference	147
8.32	workspace/S-TGDH_god/ui/Constants.java File Reference	148
8.33	workspace/S-TGDH_god/ui/CreateGroup.java File Reference	149
8.34	workspace/S-TGDH_god/ui/Interface.java File Reference	150
8.35	workspace/S-TGDH_god/ui/Manager.java File Reference	151
8.36	workspace/S-TGDH_god/ui/NodeFrame.java File Reference	152
8.37	workspace/S-TGDH_god/ui/XGridBag.java File Reference	153

Chapter 1

S-TGDH: Secure enhanced group management protocol in ad hoc networks

1.1 Introduction

1.1.1 Presentation

The S-TGDH.jar archive contains the java simulator for the S-TGDH protocol. S-TGDH is our secured extension of TGDH, TreeGroup Diffie Hellman, a group management protocol designed for ad hoc networks. In our solution, we have added stations authentication, in order to get a more secured solutions. The stations authentication only rely on a weak password, which thus made the solution adapted for common usage.

S-TGDH provides also algorithms optimisation and solve a problem of TGDH due to the distributed aspect of the solution: the group update process has been modified.

The goals of this simulator is to show that first our solution can be implemented in the reality and is not just a concept. It then also let us evaluate whether we took into account most of the specification problem for the different group management operations or not. As it is just a proof of concept, the simulator is not optimize and some choice may be the not the best one in term of algorithm optimisations / development techniques, which is however not the problems of our simulator (but they are not the goals too !).

1.1.2 information

More information about this project can be obtain by

- reading the S-TGDH article for the CRiSIS'07 submit: S-TGDH, secure enhanced group management protocol in ad~hoc networks, Frederic Cuppens, Nora Cuppens, and Julien Thomas, The International Conference on Risks and Security of Internet and Systems, CRiSIS'2007, Marrakech-Morocco 2-5 July 2007
- contacting Julien Thomas (julien.thomas@enst-bretagne.fr)
- visiting Julien Thomas research website, <http://aispirit.tuxfamily.org/~julienthomas>

- reading the manual report of the simulator, in french: Simulateur Java pour S-TGDH, available on the author's website

1.2 Utilizations

1.2.1 aspects

1.2.2 line information

As for any java archive (jar file), you must enter the `java -jar <archive>` command to execute the archive

For the S-TGDH simulator, the command line options are available with the help command.

Enter the `java -jar <archive_file> help` command to retrieve the command line options

1.2.3 Tools required:

The simulator has been developed in Java. The java runtime environment (jre) is so required. With a version superior or equal to java 1.5 (compilance level 1.5)

Chapter 2

javaSTGDH Namespace Index

2.1 javaSTGDH Namespace List

Here is a list of all documented namespaces with brief descriptions:

console (Manage the console mode of the simulator)	11
lang (Files required internationalized the simulator)	12
manager (Manage the network)	13
manager::messages (Network messages)	14
node (A test class)	15
node::buffers (A test class2)	16
node::messages (A test class2)	17
tools (A test class2)	18
ui (A test class2)	19

Chapter 3

javaSTGDH Hierarchical Index

3.1 javaSTGDH Class Hierarchy

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

console::Aleatory	23
ui::CellStyle	25
node::buffers::Challenge2Response	28
node::buffers::ChallengeExpected	30
ui::Constants	32
lang::LanguageException	63
node::DelayException	41
ui::About	21
ui::CreateGroup	35
ui::Manager	75
ui::NodeFrame	94
ui::XGridBag	113
tools::Crypto	37
manager::ILauncher	50
console::Launcher	65
ui::Launcher	69
ui::Interface	56
tools::JavaTools	58
lang::Language	61
manager::Launcher	67
manager::Log	70
manager::messages::Message	78
manager::messages::MulticastMessage	81
node::NodeBehavior	90
node::GroupNode	42
node::InitNode	51
node::NodeConstants	92
node::GroupNode	42
node::IGroupNode	47
node::INode	55
node::Node	83
node::IInitNode	49

node::InitNode	51
node::INode	55
node::INode	55
node::Node	83
node::messages::NodeMessage	95
manager::Supervisor	97
manager::SupervisorMessage	98
tools::Tree	102
tools::TreeNode	107

Chapter 4

javaSTGDH Class Index

4.1 javaSTGDH Class List

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

ui::About (About (p. 21) window of the project)	21
console::Aleatory (Manager for the console mode of the simulator)	23
ui::CellStyle (Define a cell style used by XGridBag (p. 113))	25
node::buffers::Challenge2Response (Memorize the challenge (challenge 2) to send to which node when all the nodes will have send a ANSWER_GROUP_MESSAGE)	28
node::buffers::ChallengeExpected (Memorize the challenge (challenge 1) associated to all the members of the group)	30
ui::Constants (Graphical constants for the project components)	32
ui::CreateGroup (Configuration window for a group creation)	35
tools::Crypto (Centralized the commonly used functions to perform cryptographic operations)	37
node::DelayException (Exception to manage unordered messages)	41
node::GroupNode (Model the group behavior of a node)	42
node::IGroupNode (Interface for the group behavior of a node)	47
node::IInitNode (Interface for the group initiation behavior of a node)	49
manager::ILauncher (Launcher (p. 67) interface. Each launcher must implement this interface)	50
node::InitNode (Model the group initiation behavior of a node)	51
node::INode (Model the default behavior of a node)	55
ui::Interface (Communication interface for graphical mode of the simulator)	56
tools::JavaTools (Centralized the commonly used functions to perform Java operations)	58
lang::Language (Language (p. 61) manager for the simulator)	61
lang::LanguageException (Language (p. 61) Exception)	63
console::Launcher (Launcher (p. 65) for console mode of the simulator)	65
manager::Launcher (Launcher (p. 67) of the simulator)	67
ui::Launcher (Launcher (p. 69) for graphical mode of the simulator)	69
manager::Log (Logging class)	70
ui::Manager (Core window of the simulator graphical mode)	75
manager::messages::Message (Unicast message stored in the network boxes)	78
manager::messages::MulticastMessage (Multicast message stored in the network boxes)	81
node::Node (Modelisation of a node of the network)	83
node::NodeBehavior (Model the standard behavior of a node)	90

node::NodeConstants (Interface for the nodes: contains standard constants)	92
ui::NodeFrame (Window use to retrieve information on a node of the network)	94
node::messages::NodeMessage (Modelize a message in the node stack)	95
manager::Supervisor (Network modelization)	97
manager::SupervisorMessage (Messages modelisation for the interaction between the nodes (and the network))	98
tools::Tree (Represents the cryptographic tree of the (S-)TGDH algorithm)	102
tools::TreeNode (Represents the standard element of the Tree (p.102) structure)	107
This class defines styles reusable by differents components in a grid)113	

Chapter 5

javaSTGDH File Index

5.1 javaSTGDH File List

Here is a list of all documented files with brief descriptions:

workspace/S-TGDH_god/ mainpage.doxydoc	??
workspace/S-TGDH_god/ namespace.doxydoc (Main page for the documentation) .	129
workspace/S-TGDH_god/console/ Aleatory.java (Aleatory manager for the console mode of the simulator)	117
workspace/S-TGDH_god/console/ Launcher.java (Launcher for console mode of the simulator)	118
workspace/S-TGDH_god/lang/ Language.java (Language manager for the simulator) .	121
workspace/S-TGDH_god/lang/ LanguageException.java (Language Exception) . . .	122
workspace/S-TGDH_god/manager/ ILauncher.java (Launcher interface. Each launcher must implement this interface)	123
workspace/S-TGDH_god/manager/ Launcher.java (Launcher of the simulator)	119
workspace/S-TGDH_god/manager/ Log.java (Logging class)	124
workspace/S-TGDH_god/manager/ Supervisor.java (Network modelization)	127
workspace/S-TGDH_god/manager/ SupervisorMessage.java (Messages modelisation for the interaction between the nodes (and the network))	128
workspace/S-TGDH_god/manager/messages/ Message.java (Unicast message stored in the network boxes)	125
workspace/S-TGDH_god/manager/messages/ MulticastMessage.java (Multicast message stored in the network boxes)	126
workspace/S-TGDH_god/node/ DelayException.java (Exception to manage unordered messages)	132
workspace/S-TGDH_god/node/ GroupNode.java (Model the group behavior of a node) .	133
workspace/S-TGDH_god/node/ IGroupNode.java (Interface for the group behavior of a node)	134
workspace/S-TGDH_god/node/ IInitNode.java (Interface for the group initiation behavior of a node)	135
workspace/S-TGDH_god/node/ InitNode.java (Model the group initiation behavior of a node)	136
workspace/S-TGDH_god/node/ Inode.java (Model the default behavior of a node) .	137
workspace/S-TGDH_god/node/ Node.java (Modelisation of a node of the network) .	139
workspace/S-TGDH_god/node/ NodeBehavior.java (Model the standard behavior of a node)	140

workspace/S-TGDH_god/node/ NodeConstants.java (Interface for the nodes: contains standard constants)	141
workspace/S-TGDH_god/node/buffers/ Challenge2Response.java (Memorize the challenge (challenge 2) to send to which node when all the nodes will have send a ANSWER_GROUP_MESSAGE)	130
workspace/S-TGDH_god/node/buffers/ ChallengeExpected.java (Memorize the challenge (challenge 1) associated to all the members of the group)	131
workspace/S-TGDH_god/node/messages/ NodeMessage.java (Modelize a message in the node stack)	138
workspace/S-TGDH_god/tools/ Crypto.java (Centralized the commonly used functions to perform cryptographic operations)	142
workspace/S-TGDH_god/tools/ JavaTools.java (Centralized the commonly used functions to perform Java operations)	143
workspace/S-TGDH_god/tools/ Tree.java (Represents the cryptographic tree of the (S-)TGDH algorithm)	144
workspace/S-TGDH_god/tools/ TreeNode.java (Represents the standard element of the Tree structure)	145
workspace/S-TGDH_god/ui/ About.java (About window of the project)	146
workspace/S-TGDH_god/ui/ CellStyle.java (Define a cell style used by XGridBag)	147
workspace/S-TGDH_god/ui/ Constants.java (Graphical constants for the project components)	148
workspace/S-TGDH_god/ui/ CreateGroup.java (Configuration window for a group creation)	149
workspace/S-TGDH_god/ui/ Interface.java (Communication interface for graphical mode of the simulator)	150
workspace/S-TGDH_god/ui/ Launcher.java (Launcher for graphical mode of the simulator)	120
workspace/S-TGDH_god/ui/ Manager.java (Core window of the simulator graphical mode)	151
workspace/S-TGDH_god/ui/ NodeFrame.java (Window use to retrieve information on a node of the network)	152
This class defines styles reusable by differents components in a grid)	153

Chapter 6

javaSTGDH Namespace Documentation

6.1 console Namespace Reference

Manage the console mode of the simulator.

Classes

- class **Aleatory**
Manager for the console mode of the simulator.
- class **Launcher**
Launcher(p. 65) *for console mode of the simulator.*

6.1.1 Detailed Description

Manage the console mode of the simulator.

The simulator can be launched in a console mode in which nodes' operations occurred in an aleatory way. The console package manages this console mode. To launch the simulator in console mode, see **manager.Launcher**(p. 67)

See also:

manager.Launcher::main(p. 67)(String[])

6.2 lang Namespace Reference

Files required internationalized the simulator.

Classes

- class **Language**
Language(p. 61) *manager for the simulator.*
- class **LanguageException**
Language(p. 61) *Exception.*

6.2.1 Detailed Description

Files required internationalized the simulator.

The simulator can be launched in several languages, determined by the STGDH_XX.properties files at the root of the solution. The files in the lang package manage these language files in order to use them for the simulator.

See also:

lang.Language(p. 61)

6.3 manager Namespace Reference

Manage the network.

Classes

- interface **ILauncher**
Launcher(p. 67) *interface. Each launcher must implement this interface.*
- class **Launcher**
Launcher(p. 67) *of the simulator.*
- class **Log**
Logging class.
- class **Supervisor**
Network modelization.
- interface **SupervisorMessage**
Messages modelisation for the interaction between the nodes (and the network).

Namespaces

- namespace **messages**
Network messages.

6.3.1 Detailed Description

Manage the network.

The manager package contains files such as **Supervisor.java**(p. 127) which are used to model the network.

See also:

manager.Supervisor(p. 97) for the network simulation
Log.java(p. 124) for the **messages**(p. 14) logging

/*!

6.4 manager::messages Namespace Reference

Network messages.

Classes

- class **Message**
Unicast message stored in the network boxes.
- class **MulticastMessage**
Multicast message stored in the network boxes.

6.4.1 Detailed Description

Network messages.

Contains the different kind of messages supported by the network, which for instance unicast (**Message.java**(p. 125)) and multicast (**MulticastMessage.java**(p. 126)) messages

6.5 node Namespace Reference

A test class.

Classes

- class **DelayException**
Exception to manage unordered messages.
- class **GroupNode**
Model the group behavior of a node.
- interface **IGroupNode**
Interface for the group behavior of a node.
- interface **IInitNode**
Interface for the group initiation behavior of a node.
- class **InitNode**
Model the group initiation behavior of a node.
- interface **INode**
Model the default behavior of a node.
- class **Node**
Modelisation of a node of the network.
- class **NodeBehavior**
Model the standard behavior of a node.
- interface **NodeConstants**
Interface for the nodes: contains standard constants.

Namespaces

- namespace **buffers**
A test class2.
- namespace **messages**
A test class2.

6.5.1 Detailed Description

A test class.

A more detailed class description.

/*!

6.6 node::buffers Namespace Reference

A test class2.

Classes

- class **Challenge2Response**
Memorize the challenge (challenge 2) to send to which node when all the nodes will have send a ANSWER_GROUP_MESSAGE.
- class **ChallengeExpected**
Memorize the challenge (challenge 1) associated to all the members of the group.

6.6.1 Detailed Description

A test class2.

A more detailed class description.

6.7 node::messages Namespace Reference

A test class2.

Classes

- class **NodeMessage**
Modelize a message in the node stack.

6.7.1 Detailed Description

A test class2.

A more detailed class description.

6.8 tools Namespace Reference

A test class2.

Classes

- class **Crypto**
Centralized the commonly used functions to perform cryptographic operations.
- class **JavaTools**
Centralized the commonly used functions to perform Java operations.
- class **Tree**
Represents the cryptographic tree of the (S-)TGDH algorithm.
- class **TreeNode**
*Represents the standard element of the **Tree**(p.102) structure.*

6.8.1 Detailed Description

A test class2.

A more detailed class description.

6.9 ui Namespace Reference

A test class2.

Classes

- class **About**
About(p. 21) *window of the project.*
- class **CellStyle**
Define a cell style used by XGridBag(p. 113).
- interface **Constants**
Graphical constants for the project components.
- class **CreateGroup**
Configuration window for a group creation.
- class **Interface**
Communication interface for graphical mode of the simulator.
- class **Launcher**
Launcher(p. 69) *for graphical mode of the simulator.*
- class **Manager**
Core window of the simulator graphical mode.
- class **NodeFrame**
Window use to retrieve information on a node of the network.
- class **XGridBag**
The XGridBag(p. 113) *helps to use the GridBagConstraints.*
This class defines styles reusable by differents components in a grid.

6.9.1 Detailed Description

A test class2.

A more detailed class description.

Chapter 7

javaSTGDH Class Documentation

7.1 ui::About Class Reference

About(p. 21) window of the project.

Inheritance diagram for ui::About::

Public Member Functions

- **About** (String filename)

Package Attributes

- JTextPane **htmlEditor**

7.1.1 Detailed Description

About(p. 21) window of the project.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition at line 25 of file About.java.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 ui::About::About (String *filename*) [inline]

Create and display an **About**(p. 21) window

Parameters:

filename the html file describing the project

Definition at line 40 of file About.java.

7.1.3 Member Data Documentation

7.1.3.1 JTextPane ui::About::htmlEditor [package]

The component which contains the html description

Definition at line 29 of file About.java.

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

- workspace/S-TGDH_god/ui/**About.java**

7.2 console::Aleatory Class Reference

Manager for the console mode of the simulator.

Public Member Functions

- String **getLogExtension** ()
- **Aleatory** (**Supervisor** *s*, long *timeToRun*)
- void **init** ()
- Override void **start** ()
- boolean **checkGroupInfos** ()

7.2.1 Detailed Description

Manager for the console mode of the simulator.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition at line 35 of file Aleatory.java.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 console::Aleatory::Aleatory (**Supervisor** *s*, long *timeToRun*) [inline]

Initiate the aleatory mode Need to call **start()**(p. 24) after

Parameters:

s the handle to the network

timeToRun the time to perform the simulation, in ms

Definition at line 71 of file Aleatory.java.

7.2.3 Member Function Documentation

7.2.3.1 boolean console::Aleatory::checkGroupInfos () [inline]

Perform statistics checks & analysis

Returns:

true if all the tests succeed

Definition at line 261 of file Aleatory.java.

7.2.3.2 String console::Aleatory::getLogExtension () [inline]

return the extension name of the files in process Files are log.txt_extension and error.log.txt_extension

Returns:

the extension

Definition at line 61 of file Aleatory.java.

7.2.3.3 void console::Aleatory::init () [inline]

Display basic informations on the console The console needs to support ASCII characters

Definition at line 119 of file Aleatory.java.

7.2.3.4 Override void console::Aleatory::start () [inline]

Launch the aleatory simulator operations (temporized process)

Definition at line 151 of file Aleatory.java.

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

- workspace/S-TGDH_god/console/**Aleatory.java**

7.3 ui::CellStyle Class Reference

Define a cell style used by `XGridBag`(p. 113).

Public Member Functions

- `CellStyle` (double `weightx`, double `weighty`, int `anchor`, int `fill`, Insets `insets`, int `ipadx`, int `ipady`)

Package Attributes

- int `anchor`
- int `fill`
- Insets `insets`
- int `ipadx`
- int `ipady`
- double `weightx`
- double `weighty`

7.3.1 Detailed Description

Define a cell style used by `XGridBag`(p. 113).

Date:

2007/06/13

See also:

`XGridBag`(p. 113)
`java.awt.GridBagConstraints`

Version:

1.0

Author:

//Web source//

Definition at line 22 of file `CellStyle.java`.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 ui::CellStyle::CellStyle (double *weightx*, double *weighty*, int *anchor*, int *fill*, Insets *insets*, int *ipadx*, int *ipady*) [inline]

`CellStyle`(p. 25) constructor.

Parameters:

weightx extra X space usage (none=0.0 ... all=1.0)

weighty extra Y space usage (none=0.0 ... all=1.0)

anchor location if component doesn't occupy entire cell:

```

-----
|FIRST_LINE_START...PAGE_START.....FIRST_LINE_END|
|LINE_START.....CENTER.....LINE_END|
|LAST_LINE_START.....PAGE_END.....LAST_LINE_END|
-----

```

fill NONE, HORIZONTAL, VERTICAL, BOTH

insets (int top, int left, int bottom, int right)

ipadx internal X padding

ipady internal Y padding

See also:

XGridBag(p. 113)

java.awt.GridBagConstraints

java.awt.Insets

Definition at line 81 of file CellStyle.java.

7.3.3 Member Data Documentation

7.3.3.1 int ui::CellStyle::anchor [package]

anchor CENTER, NORTH, NORTHEAST, EAST, SOUTHEAST, SOUTH, SOUTHWEST, WEST, NORTHWEST.

Definition at line 27 of file CellStyle.java.

7.3.3.2 int ui::CellStyle::fill [package]

fill NONE, HORIZONTAL, VERTICAL, BOTH

Definition at line 32 of file CellStyle.java.

7.3.3.3 Insets ui::CellStyle::insets [package]

insets (int top, int left, int bottom, int right)

Definition at line 37 of file CellStyle.java.

7.3.3.4 int ui::CellStyle::ipadx [package]

internal X padding

Definition at line 42 of file CellStyle.java.

7.3.3.5 int ui::CellStyle::ipady [package]

internal Y padding

Definition at line 47 of file CellStyle.java.

7.3.3.6 double ui::CellStyle::weightx [package]

extra X space (none=0.0 ... all=1.0)

Definition at line 52 of file CellStyle.java.

7.3.3.7 double ui::CellStyle::weighty [package]

extra Y space (none=0.0 ... all=1.0)

Definition at line 57 of file CellStyle.java.

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

- workspace/S-TGDH_god/ui/CellStyle.java

7.4 node::buffers::Challenge2Response Class Reference

Memorize the challenge (challenge 2) to send to which node when all the nodes will have send a ANSWER_GROUP_MESSAGE.

Public Member Functions

- **Challenge2Response** (String dest, String m)
- String **getMessage** ()
- String **getNodeDest** ()

7.4.1 Detailed Description

Memorize the challenge (challenge 2) to send to which node when all the nodes will have send a ANSWER_GROUP_MESSAGE.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 16 of file Challenge2Response.java.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 node::buffers::Challenge2Response::Challenge2Response (String *dest*, String *m*) [inline]

Constructor

Parameters:

dest node identifier

m associated message

Definition at line 33 of file Challenge2Response.java.

7.4.3 Member Function Documentation

7.4.3.1 String node::buffers::Challenge2Response::getMessage () [inline]

Return the message to send

Returns:

Definition at line 42 of file Challenge2Response.java.

7.4.3.2 String node::buffers::Challenge2Response::getNodeDest () [inline]

Return the node identifier

Returns:

Definition at line 50 of file Challenge2Response.java.

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

- workspace/S-TGDH_god/node/buffers/**Challenge2Response.java**

7.5 node::buffers::ChallengeExpected Class Reference

Memorize the challenge (challenge 1) associated to all the members of the group.

Public Member Functions

- **ChallengeExpected** (String nodeID, String challenge)
- String **getChallenge** ()
- String **getNodeID** ()

7.5.1 Detailed Description

Memorize the challenge (challenge 1) associated to all the members of the group.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 16 of file ChallengeExpected.java.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 node::buffers::ChallengeExpected::ChallengeExpected (String *nodeID*, String *challenge*) [inline]

Constructor

Parameters:

nodeID the node identifier

challenge the associated challenge

Definition at line 33 of file ChallengeExpected.java.

7.5.3 Member Function Documentation

7.5.3.1 String node::buffers::ChallengeExpected::getChallenge () [inline]

Return the challenge value

Returns:

Definition at line 42 of file ChallengeExpected.java.

7.5.3.2 String node::buffers::ChallengeExpected::getNodeID () [inline]

Return the node identifier

Returns:

Definition at line 50 of file ChallengeExpected.java.

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

- workspace/S-TGDH_god/node/buffers/**ChallengeExpected.java**

7.6 ui::Constants Interface Reference

Graphical constants for the project components.

Inheritance diagram for ui::Constants::

Static Public Attributes

- static final int **CREATE_GROUP_WIDTH** = 300
- static final int **CREATE_GROUP_HEIGHT** = 300
- static final int **MANAGER_WIDTH** = 350
- static final int **MANAGER_HEIGHT** = 420
- static final int **NODE_MESSAGE_WIDTH** = 300
- static final int **NODE_MESSAGE_HEIGHT** = 400
- static final int **ABOUT_WIDTH** = 400
- static final int **ABOUT_HEIGHT** = 500
- static final long **PROJECTUID** = 1234567
- static final int **DELAYEXCEPTION_SUBID** = 1
- static final int **LANGUAGEEXCEPTION_SUBID** = 2
- static final int **ABOUT_SUBID** = 3
- static final int **CREATEGROUP_SUBID** = 4
- static final int **MANAGER_SUBID** = 5
- static final int **NODEFRAME_SUBID** = 6
- static final int **XGRIDBAG_SUBID** = 7

7.6.1 Detailed Description

Graphical constants for the project components.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition at line 16 of file Constants.java.

7.6.2 Member Data Documentation

7.6.2.1 final int ui::Constants::ABOUT_HEIGHT = 500 [static]

About(p. 21) window height

Definition at line 58 of file Constants.java.

7.6.2.2 final int ui::Constants::ABOUT_SUBID = 3 [static]

About(p. 21) class identifier

Definition at line 81 of file Constants.java.

7.6.2.3 final int ui::Constants::ABOUT_WIDTH = 400 [static]

About(p. 21) window width

Definition at line 53 of file Constants.java.

7.6.2.4 final int ui::Constants::CREATE_GROUP_HEIGHT = 300 [static]

CreateGroup(p. 35) window height

Definition at line 28 of file Constants.java.

7.6.2.5 final int ui::Constants::CREATE_GROUP_WIDTH = 300 [static]

CreateGroup(p. 35) window width

Definition at line 23 of file Constants.java.

7.6.2.6 final int ui::Constants::CREATEGROUP_SUBID = 4 [static]

DelayException class identifier

Definition at line 86 of file Constants.java.

7.6.2.7 final int ui::Constants::DELAYEXCEPTION_SUBID = 1 [static]

DelayException class identifier

Definition at line 71 of file Constants.java.

7.6.2.8 final int ui::Constants::LANGUAGEEXCEPTION_SUBID = 2 [static]

LanguageException class identifier

Definition at line 76 of file Constants.java.

7.6.2.9 final int ui::Constants::MANAGER_HEIGHT = 420 [static]

Manager(p. 75) window height

Definition at line 38 of file Constants.java.

7.6.2.10 final int ui::Constants::MANAGER_SUBID = 5 [static]

Manager(p. 75) class identifier

Definition at line 91 of file Constants.java.

7.6.2.11 `final int ui::Constants::MANAGER_WIDTH = 350 [static]`

Manager(p. 75) window width

Definition at line 33 of file Constants.java.

7.6.2.12 `final int ui::Constants::NODE_MESSAGE_HEIGHT = 400 [static]`

NodeMessage window height

Definition at line 48 of file Constants.java.

7.6.2.13 `final int ui::Constants::NODE_MESSAGE_WIDTH = 300 [static]`

NodeMessage window width

Definition at line 43 of file Constants.java.

7.6.2.14 `final int ui::Constants::NODEFRAME_SUBID = 6 [static]`

NodeFrame(p. 94) class identifier

Definition at line 96 of file Constants.java.

7.6.2.15 `final long ui::Constants::PROJECTUID = 1234567 [static]`

project main identifier

Definition at line 66 of file Constants.java.

7.6.2.16 `final int ui::Constants::XGRIDBAG_SUBID = 7 [static]`

XGridBag(p. 113) class identifier

Definition at line 101 of file Constants.java.

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

- workspace/S-TGDH_god/ui/Constants.java

7.7 ui::CreateGroup Class Reference

Configuration window for a group creation.

Inheritance diagram for ui::CreateGroup::

Public Member Functions

- **CreateGroup** (**Manager** manager)
- void **addHandlers** ()
- void **init** (List< **INode** > list)

7.7.1 Detailed Description

Configuration window for a group creation.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition at line 38 of file CreateGroup.java.

7.7.2 Constructor & Destructor Documentation

7.7.2.1 ui::CreateGroup::CreateGroup (**Manager** *manager*) [inline]

Create the configuration window

Parameters:

manager handler of the main window of the simulator

See also:

addHandlers()(p. 35)

Definition at line 83 of file CreateGroup.java.

7.7.3 Member Function Documentation

7.7.3.1 void ui::CreateGroup::addHandlers () [inline]

Add the action handlers to the different buttons

Definition at line 143 of file CreateGroup.java.

7.7.3.2 void ui::CreateGroup::init (List< INode > *list*) [inline]

Window initialisation

Parameters:

list the list of the existing nodes in the network

Definition at line 181 of file CreateGroup.java.

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

- workspace/S-TGDH_god/ui/**CreateGroup.java**

7.8 tools::Crypto Class Reference

Centralized the commonly used functions to perform cryptographic operations.

Static Public Member Functions

- static long **buildK** ()
- static long **buildGK** (long privateK)
- static long **buildParentPrivateKey** (long privateK, long neighb_GK)
- static String **cryptMessage** (long key, String message)
- static String **decryptMessage** (long key, String message)
- static long[] **buildRSAKeys** ()
- static String **cryptRSAMessage** (long key, String message)
- static String **decryptRSAMessage** (long key, String message)
- static String **hashMessage** (String message)
- static void **main** (String[] args)

Static Public Attributes

- static final int **ALPHA** = 5

7.8.1 Detailed Description

Centralized the commonly used functions to perform cryptographic operations.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Note:

functions may be correct functions in the sense of behavior are simulation. For instance, the RSAoperations are just abstract operations: $\text{decryp}(\text{crypt}(c)) = c$ but $\text{crypt}(c)$ is not the RSA exact function.

Definition at line 22 of file Crypto.java.

7.8.2 Member Function Documentation

7.8.2.1 static long tools::Crypto::buildGK (long *privateK*) [inline, static]

Build the public GK key based on the private key K.

Parameters:

privateK the private key

Returns:

the public key

Definition at line 99 of file Crypto.java.

7.8.2.2 static long tools::Crypto::buildK () [inline, static]

Generate a K key, considered as the Diffie-Hellman private key in S-TGDH.

Note:

The function last at least 5 ms, in order to prevent two simultaneous called to **buildK()**(p. 38), which would produce the same private key

Returns:

a new K key

Definition at line 44 of file Crypto.java.

7.8.2.3 static long tools::Crypto::buildParentPrivateKey (long *privateK*, long *neighb_GK*) [inline, static]

Evaluate the parent private key based on its private key and the neighbour public key.

Parameters:

privateK the node private key

neighb_GK the neighbour public key

Returns:

the parent private key

Definition at line 112 of file Crypto.java.

7.8.2.4 static long [] tools::Crypto::buildRSAKeys () [inline, static]

Generates a new couple of RSA keys.

Returns:

the keys as {private, public}

Definition at line 206 of file Crypto.java.

7.8.2.5 static String tools::Crypto::cryptMessage (long *key*, String *message*) [inline, static]

Cypher a message - symetric encryption

Parameters:

key the encryption key

message the text to cypher

Returns:

the cyphered text

Definition at line 158 of file Crypto.java.

7.8.2.6 static String tools::Crypto::cryptRSAMessage (long *key*, String *message*)
[inline, static]

RSA encryption.

Note:

cyphering: key=dest_public_key or signature: key=our_private_key
This is an RSA encryption emulation

Parameters:

key encryption key
message message to cypher

Returns:

RSA encrypted version

Definition at line 225 of file Crypto.java.

7.8.2.7 static String tools::Crypto::decryptMessage (long *key*, String *message*)
[inline, static]

Decypher a message - symetric encryption

Note:

decryptMessage = cryptMessage as the encryption process is symetric

Parameters:

key the deciphering key
message the text to decypher

Returns:

the clear text

Definition at line 198 of file Crypto.java.

7.8.2.8 static String tools::Crypto::decryptRSAMessage (long *key*, String *message*)
[inline, static]

RSA decryption.

Note:

decyphering: key=our_private_key or signature check: key=sender_public_key
This is an RSA decryption emulation

Parameters:

key decryption key
message message to decypher

Returns:

clear

Definition at line 239 of file Crypto.java.

7.8.2.9 `static String tools::Crypto::hashMessage (String message)` [inline, static]

Generates a hash of the message.

Parameters:

message message to hash

Returns:

the hash version of the message

Definition at line 248 of file Crypto.java.

7.8.2.10 `static void tools::Crypto::main (String[] args)` [inline, static]

Conventional test function

Parameters:

args

Definition at line 258 of file Crypto.java.

7.8.3 Member Data Documentation

7.8.3.1 `final int tools::Crypto::ALPHA = 5` [static]

Diffie-Hellman configuration (exponent base).

Note:

the maximal bound is `(long) Math.sqrt(Long.MAX_VALUE / 2);`

Definition at line 28 of file Crypto.java.

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

- `workspace/S-TGDH_god/tools/Crypto.java`

7.9 node::DelayException Class Reference

Exception to manage unordered messages.

Inheritance diagram for node::DelayException::

Static Package Attributes

- static final long **serialVersionUID** = **PROJECTUID** + **DELAYEXCEPTION_**
SUBID

7.9.1 Detailed Description

Exception to manage unordered messages.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 17 of file DelayException.java.

7.9.2 Member Data Documentation

7.9.2.1 final long node::DelayException::serialVersionUID = PROJECTUID + DELAYEXCEPTION_ SUBID [static, package]

Object identifier

Definition at line 21 of file DelayException.java.

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

- workspace/S-TGDH_god/node/**DelayException.java**

7.10 node::GroupNode Class Reference

Model the group behavior of a node.

Inheritance diagram for node::GroupNode::

Public Member Functions

- List< String > **getNodesList** ()
- **GroupNode** (**Node** n, List< String > **nodesList**)
- void **init** ()
- synchronized void **initFromSponsor** (**INode** sponsor)
- synchronized void **doReceivedGKMessage** (String nodeID, String message) throws DelayException
- synchronized void **doParentKMessage** (String _K)
- void **intro** ()
- synchronized void **doAddNodeMessage** (String nodeSponsor, String message) throws DelayException
- synchronized boolean **doRemoveNodeMessage** (String nodeSponsor, String message) throws DelayException
- synchronized String **addNodeSponsor** (**INode** node, long node_GK)
- synchronized boolean **updateKeyBeforeAdd** (String node2addID)
- int **canAddNewNode** ()
- synchronized boolean **removeFromGroup** ()
- synchronized boolean **removeFromGroupConfirm** ()
- synchronized boolean **removeNodeSponsor** (String node2remove, String message)

Protected Attributes

- List< String > **nodesList**
- long **K**
- long **GK**
- long **GROUP_KEY**
- **Tree** **cryptoTree**

7.10.1 Detailed Description

Model the group behavior of a node.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 28 of file GroupNode.java.

7.10.2 Constructor & Destructor Documentation

7.10.2.1 node::GroupNode::GroupNode (Node *n*, List< String > *nodesList*) [inline]

Constructeur

Parameters:

- n* current node
- nodesList* all the nodes in the group

Definition at line 81 of file GroupNode.java.

7.10.3 Member Function Documentation

7.10.3.1 synchronized String node::GroupNode::addNodeSponsor (INode *node*, long *node_GK*) [inline]

Add the node to the group, by acting as it's sponsor Note that this methods his supposed to simulate the cooperation between the sponsor and the node to be added, like

See also:

[initFromSponsor\(INode\)](#)(p. 45)

Parameters:

- node* the node to be added
- node_GK* the new node GK key

Returns:

null if everything goes well, the error message otherwise

Definition at line 770 of file GroupNode.java.

7.10.3.2 int node::GroupNode::canAddNewNode () [inline]

The node support the entry of a new node

Returns:

-1: node is not a group node, -2: node is not a leaf -3: the tree would not be balanced and 1: authorized

Definition at line 874 of file GroupNode.java.

7.10.3.3 synchronized void node::GroupNode::doAddNodeMessage (String *nodeSponsor*, String *message*) throws DelayException [inline]

Upon receiving a AddNode message ...

Parameters:

- nodeSponsor*
- message* In this version, message = GK, previousRootToReplace, newNodeID, newSubtree

Definition at line 614 of file GroupNode.java.

7.10.3.4 `synchronized void node::GroupNode::doParentKMessage (String _K)` [inline]

The current node receive his private key from one of his child (to be precise, the first left child which is not virtual) It has to broadcast the associated global key to the whole group

Parameters:

`_K` the private key

Note:

no check is done about "is the child node the good one"

Definition at line 473 of file GroupNode.java.

7.10.3.5 `synchronized void node::GroupNode::doReceivedGKMessage (String nodeID, String message) throws DelayException` [inline]

Receive a GK key

Parameters:

`nodeID` the node which has emitted the key

`message` structure : nodeID, GK

Definition at line 210 of file GroupNode.java.

7.10.3.6 `synchronized boolean node::GroupNode::doRemoveNodeMessage (String nodeSponsor, String message) throws DelayException` [inline]

Upon receiving a RemoveNode message ...

Parameters:

`nodeSponsor`

`message`

Returns:

Definition at line 663 of file GroupNode.java.

7.10.3.7 `List<String> node::GroupNode::getNodesList ()` [inline]

Return the list of the nodes in the group

Definition at line 72 of file GroupNode.java.

7.10.3.8 `void node::GroupNode::init ()` [inline, virtual]

When the group is created, the method `init` is supposed to be called by all the members It build all the needed informations and send messages if needed

Implements `node::NodeBehavior` (p. 90).

Definition at line 99 of file GroupNode.java.

7.10.3.9 synchronized void node::GroupNode::initFromSponsor (INode *sponsor*) [inline]

The node to be added to the group needs the intervention of a node inside the group, its sponsor

The standard behavior is : + sponsor.updateKeyBeforeAdd + newNode.joinGroup() + newNode.initFromSponsor() <- retrieve the current information of the group, after key updates ! + sponsor.addNode2Group() <- tell the other members of the group that a node is to be added

Parameters:

sponsor the sponsor node

Definition at line 145 of file GroupNode.java.

7.10.3.10 void node::GroupNode::intro () [inline]

Messages to be displayed in the logs

Definition at line 580 of file GroupNode.java.

7.10.3.11 synchronized boolean node::GroupNode::removeFromGroup () [inline]

The current node performs operations to leave the group Not that the node has to wait confirmation from its sponsor.

See also:

removeFromGroupConfirm()(p. 45) Otherwise the operation is canceled

Returns:

true if the leaving initiation succeeded

Definition at line 897 of file GroupNode.java.

7.10.3.12 synchronized boolean node::GroupNode::removeFromGroupConfirm () [inline]

Upon receiving a confirm answer from the sponsor ... I leave

Definition at line 961 of file GroupNode.java.

7.10.3.13 synchronized boolean node::GroupNode::removeNodeSponsor (String *node2remove*, String *message*) [inline]

Receive a removing message (ask for sponsor) Performs check before broadcasting the message : is the signature valid ? is this node the sponsor of the one to remove ?

Parameters:

node2remove the ID of the node to be removed

message contains the proof of the node to leave (c, sign(c))

Returns:

true if accept the departing

Definition at line 979 of file GroupNode.java.

7.10.3.14 synchronized boolean `node::GroupNode::updateKeyBeforeAdd (String node2addID)` [inline]

The current node, which will act as sponsor, needs to rebuild its keys. The standard behavior is : + `sponsor.updateKeyBeforeAdd` + `newNode.joinGroup()` + `newNode.initFromSponsor()` <- retrieve the current information of the group, after key updates ! + `sponsor.addNode2Group()` <- tell the other members of the group that a node is to be added

Parameters:

node2addID identifier of the node to add

Returns:

true if everything goes well

Definition at line 850 of file `GroupNode.java`.

7.10.4 Member Data Documentation

7.10.4.1 Tree `node::GroupNode::cryptoTree` [protected]

TGDH information: cryptographic tree

Definition at line 55 of file `GroupNode.java`.

7.10.4.2 long `node::GroupNode::GK` [protected]

TGDH information: public key GK

Definition at line 45 of file `GroupNode.java`.

7.10.4.3 long `node::GroupNode::GROUP_KEY` [protected]

TGDH information: group private key

Definition at line 50 of file `GroupNode.java`.

7.10.4.4 long `node::GroupNode::K` [protected]

TGDH information: private key K

Definition at line 40 of file `GroupNode.java`.

7.10.4.5 `List<String> node::GroupNode::nodesList` [protected]

list of the nodes in the group

Definition at line 33 of file `GroupNode.java`.

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

- `workspace/S-TGDH_god/node/GroupNode.java`

7.11 node::IGroupNode Interface Reference

Interface for the group behavior of a node.

Inheritance diagram for node::IGroupNode::

Public Member Functions

- int **canAddNewNode** ()
- boolean **updateKeyBeforeAdd** (String node2addID)
- boolean **removeFromGroup** ()

7.11.1 Detailed Description

Interface for the group behavior of a node.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 17 of file IGroupNode.java.

7.11.2 Member Function Documentation

7.11.2.1 int node::IGroupNode::canAddNewNode ()

The node support the entry of a new node

Returns:

-1: node is not a group node, -2: node is not a leaf -3: the tree would not be balanced and 1: authorized

Implemented in **node::Node** (p. 85).

7.11.2.2 boolean node::IGroupNode::removeFromGroup ()

The node wants to leave the group

Implemented in **node::Node** (p. 87).

7.11.2.3 boolean `node::IGroupNode::updateKeyBeforeAdd` (String *node2addID*)

The current node, which will act as sponsor, needs to rebuild its keys.

Parameters:

node2addID identifier of the node to add

Returns:

true if everything goes well

Implemented in `node::Node` (p. 88).

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

- `workspace/S-TGDH_god/node/IGroupNode.java`

7.12 node::IInitNode Interface Reference

Interface for the group initiation behavior of a node.

Inheritance diagram for node::IInitNode::

Public Member Functions

- void **initGroup** (List< String > nodeListID)

7.12.1 Detailed Description

Interface for the group initiation behavior of a node.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 18 of file IInitNode.java.

7.12.2 Member Function Documentation

7.12.2.1 void node::IInitNode::initGroup (List< String > *nodeListID*)

The node acts as the group initiator

Parameters:

nodeListID list of the potential nodes for the group

Implemented in **node::InitNode** (p. 53), and **node::Node** (p. 86).

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

- workspace/S-TGDH_god/node/IInitNode.java

7.13 manager::ILauncher Interface Reference

Launcher(p. 67) interface. Each launcher must implement this interface.

Inheritance diagram for manager::ILauncher::

Public Member Functions

- void **launch** (String[] args)

7.13.1 Detailed Description

Launcher(p. 67) interface. Each launcher must implement this interface.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition at line 16 of file ILauncher.java.

7.13.2 Member Function Documentation

7.13.2.1 void manager::ILauncher::launch (String[] args)

Classical entry point: start the launcher

Implemented in **console::Launcher** (p. 66), and **ui::Launcher** (p. 69).

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

- workspace/S-TGDH_god/manager/**ILauncher.java**

7.14 node::InitNode Class Reference

Model the group initiation behavior of a node.

Inheritance diagram for node::InitNode::

Public Member Functions

- **InitNode** (**Node** *main*, boolean *groupSrc*)
- Override void **init** ()
- void **setSource** (boolean *value*)
- void **initGroup** (List< String > *nodesID*)
- boolean **answerInit** (String *nodeID*, String *mess*)
- boolean **cancelInitGroup** (String *nodeID*, String *reason*, boolean *checked*)
- synchronized boolean **confirmInit** (String *nodeID*, String *mess*)
- boolean **finishInit** (String *nodeID*, String *mess*)

Public Attributes

- ArrayList< **ChallengeExpected** > **initNodes**
- **ChallengeExpected** **initGroup**
- int **nbNodeConclude** = 0

Protected Attributes

- int **group_password**

7.14.1 Detailed Description

Model the group initiation behavior of a node.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 36 of file InitNode.java.

7.14.2 Constructor & Destructor Documentation

7.14.2.1 node::InitNode::InitNode (**Node** *main*, boolean *groupSrc*) [inline]

Constructor

Parameters:*main* Handle on the associated node*groupSrc* determine whether the current node is the node initiator or not

Definition at line 70 of file InitNode.java.

7.14.3 Member Function Documentation**7.14.3.1 boolean node::InitNode::answerInit (String *nodeID*, String *mess*)**
[inline]

The node send a ANSWER_INIT_GROUP message to the source (upon receiving a INIT_GROUP message from it)

Parameters:*nodeID**mess***Returns:**

true if the operation worked correctly

Definition at line 147 of file InitNode.java.

7.14.3.2 boolean node::InitNode::cancelInitGroup (String *nodeID*, String *reason*, boolean *checked*) [inline]

Cancel the group establishment Note : here, the node identity has been proved (

See also:**Node**(p. 83)) The remaining step is to check if this **node**(p. 15) belongs to the group. the associated behavior depends of whether the current **node**(p. 15) is the initiator or not.**Parameters:***nodeID* the node which sends the CANCEL_GROUP_MESSAGE*reason* the reason telling why the group is cancelled*checked* whether the signature message has been or not**Returns:**

true if the operation worked correctly

Definition at line 199 of file InitNode.java.

7.14.3.3 synchronized boolean node::InitNode::confirmInit (String *nodeID*, String *mess*) [inline]

The initiator node validates the nodeID authentication (challenge 1) When all the nodes send a ANSWER_INIT_GROUP to the initiator node, it sends a FINISH_INIT_GROUP

Parameters:*nodeID*

mess

Returns:

true if the operation worked correctly

Note:

several optimisation could be made for a real life protocol : the timer (marked by [AT]OPTIMIZE) could be calculated to either stop the protocol after an estimated time. It could be also calculated to resend the FINISH_INIT_GROUP packet (message lost or more probably intermediate bad node)

Definition at line 264 of file InitNode.java.

7.14.3.4 `boolean node::InitNode::finishInit (String nodeID, String mess)` [inline]

Nodes validate the source authentication (challenge 2) Then, emit a FINISH_INIT_GROUP to the initiator or a CANCEL_GROUP_MESSAGE

Parameters:

nodeID

mess

Returns:

true if the operation worked corectly

Note:

in order to prevent group establishment problems due to adresss spoofing, a CANCEL_GROUP_MESSAGE should be send not after the challenge failed, but after an amount of time has been elapsed

Definition at line 349 of file InitNode.java.

7.14.3.5 `Override void node::InitNode::init ()` [inline, virtual]

See also:

`NodeBehavior`(p. 90)

Implements `node::NodeBehavior` (p. 90).

Definition at line 82 of file InitNode.java.

7.14.3.6 `void node::InitNode::initGroup (List< String > nodesID)` [inline]

The initiator node sends a INIT_GROUP to the other members of the group

Parameters:

nodesID list of the potential nodes for the group

Implements `node::IInitNode` (p. 49).

Definition at line 116 of file InitNode.java.

7.14.3.7 void node::InitNode::setSource (boolean *value*) [inline]

Set whether the current node is the initiator of the group or not

Parameters:

value

Definition at line 104 of file InitNode.java.

7.14.4 Member Data Documentation

7.14.4.1 int node::InitNode::group_password [protected]

Weak group password

Definition at line 61 of file InitNode.java.

7.14.4.2 ChallengeExpected node::InitNode::initGroup

Challenge expected from the initiator

Definition at line 49 of file InitNode.java.

7.14.4.3 ArrayList<ChallengeExpected> node::InitNode::initNodes

List of the challenges of each node

Definition at line 43 of file InitNode.java.

7.14.4.4 int node::InitNode::nbNodeConclude = 0

Number of nodes that acknowledged

Definition at line 53 of file InitNode.java.

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

- workspace/S-TGDH_god/node/InitNode.java

7.15 node::INode Interface Reference

Model the default behavior of a node.

Inheritance diagram for node::INode::

7.15.1 Detailed Description

Model the default behavior of a node.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 22 of file INode.java.

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

- workspace/S-TGDH_god/node/INode.java

7.16 ui::Interface Class Reference

Communication interface for graphical mode of the simulator.

Static Public Member Functions

- static void **removeErrorInfo** (String *id*)
- static void **removeConfirmInfo** (String *id*)

Static Protected Member Functions

- static void **setInstance** (Manager *m*)

7.16.1 Detailed Description

Communication interface for graphical mode of the simulator.

Author:

Julien Thomas

Version:

1.0

Date:

2007/07/09

Definition at line 16 of file Interface.java.

7.16.2 Member Function Documentation

7.16.2.1 static void ui::Interface::removeConfirmInfo (String *id*) [inline, static]

Tells the graphical interface that the removal process for a node succeeded

Parameters:

id identifier of the node who initiates the remove operation

Definition at line 45 of file Interface.java.

7.16.2.2 static void ui::Interface::removeErrorInfo (String *id*) [inline, static]

Tells the graphical interface that the removal process for a node failed

Parameters:

id identifier of the node who initiates the remove operation

Definition at line 36 of file Interface.java.

7.16.2.3 `static void ui::Interface::setInstance (Manager m)` [inline, static, protected]

Assign a value to the graphical interface handle

Parameters:

m graphical interface link

Definition at line 28 of file Interface.java.

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

- workspace/S-TGDH_god/ui/**Interface.java**

7.17 tools::JavaTools Class Reference

Centralized the commonly used functions to perform Java operations.

Static Public Member Functions

- static final synchronized long **readLong** ()
- static final synchronized int **readInt** ()
- static List< String > **clone** (List< String > l)
- static List< **NodeMessage** > **cloneNodeMessages** (List< **NodeMessage** > l)
- static boolean **equals** (List< String > l, List< String > l2)
- static boolean **compare** (**Tree** t1, **Tree** t2)
- static boolean **isSubList** (List subList, List mainList)

7.17.1 Detailed Description

Centralized the commonly used functions to perform Java operations.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition at line 25 of file JavaTools.java.

7.17.2 Member Function Documentation

7.17.2.1 static List<String> tools::JavaTools::clone (List< String > l) [inline, static]

Performed a clone of the String list in parameter

Parameters:

l the list to the clone

Returns:

the copy of the list in parameter

Definition at line 98 of file JavaTools.java.

7.17.2.2 static List<NodeMessage> tools::JavaTools::cloneNodeMessages (List< NodeMessage > l) [inline, static]

Performed a clone of the NodeMessage list in parameter

Parameters:

l the list to the clone

Returns:

the copy of the list in parameter

Definition at line 114 of file JavaTools.java.

7.17.2.3 static boolean tools::JavaTools::compare (Tree *t1*, Tree *t2*) [inline, static]

Compare the structure of the 2 trees. Which means : node ID and childs recursive compare

Parameters:

t1 the first **Tree**(p. 102) to compare

t2 the second **Tree**(p. 102) to compare

Returns:

true if the tree are equals

Definition at line 152 of file JavaTools.java.

7.17.2.4 static boolean tools::JavaTools::equals (List< String > *l*, List< String > *l2*) [inline, static]

Compare two String list

Parameters:

l the first list to compare

l2 the second list to compare

Returns:

true if the list are equals (contains the same String elements)

Definition at line 131 of file JavaTools.java.

7.17.2.5 static boolean tools::JavaTools::isSubList (List *subList*, List *mainList*) [inline, static]

Compare two lists and check whether the first list is a sublist of the second one

Parameters:

subList the considered sublist

mainList the considered parent list

Returns:

true if the first list is a sublist of the second one

Definition at line 185 of file JavaTools.java.

7.17.2.6 static final synchronized int tools::JavaTools::readInt () [inline, static]

Read an int typed in the console

Returns:

the int typed by the user

Definition at line 64 of file JavaTools.java.

7.17.2.7 static final synchronized long tools::JavaTools::readLong () [inline, static]

Read a long typed in the console

Returns:

the long typed by the user

Definition at line 31 of file JavaTools.java.

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

- workspace/S-TGDH_god/tools/**JavaTools.java**

7.18 lang::Language Class Reference

Language(p. 61) manager for the simulator.

Public Member Functions

- **Language** (String language, boolean force) throws LanguageException

Static Public Attributes

- static String **UNEXPECTED_ERROR**
- static String **BRUTAL_EXIT**
- static String **SECONDBRUTAL_EXIT**
- static String **ABOUT**
- static String **UNABLE_TO_READ_FILE**
- static String **GROUP_CREATION**
- static String **TO_CONFIRM**
- static String **TO_CANCEL**
- static String **ACTIONS**
- static String **PASSWORDCREATE**
- static String **PASSWORD**
- static String **INIT_SELECT**
- static String **INITIATOR**
- static String **NODES_SELECT**
- static String **SIMULATOR_TITLE**
- static String **ADD_TITLE**
- static String **ADD_NODE**
- static String **ADD_SPONSOR**
- static String **REMOVE_TITLE**
- static String **ADD_A_NODE**
- static String **NODES_NUMBER**
- static String **TERMINATE**
- static String **TERMINATED**
- static String **DISPLAY_NODE**
- static String **TO_REMOVE**
- static String **TO_ADD**
- static String **EAVESDROPPING**
- static String **GROUP_INFOS**
- static String **NODES_MESSAGES**
- static String **REINIT**
- static String **GROUP_MANAGEMENT**
- static String **MESSAGES_FROM**
- static String **MESSAGES**
- static String **CREATE_A_GROUP**
- static String **REMOVECONFIRM**
- static String **REMOVEERROR**

7.18.1 Detailed Description

Language(p. 61) manager for the simulator.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition at line 19 of file Language.java.

7.18.2 Constructor & Destructor Documentation

7.18.2.1 lang::Language::Language (String *language*, boolean *force*) throws LanguageException [inline]

Parameters:

language language for the simulator

force

Exceptions:

LanguageException(p. 63)

Note:

the language supported are link to the .properties files. For instance, STGDH_fr.properties
=> fr (french) mode is available

Definition at line 67 of file Language.java.

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

- workspace/S-TGDH_god/lang/**Language.java**

7.19 lang::LanguageException Class Reference

Language(p. 61) Exception.

Inheritance diagram for lang::LanguageException::

Public Member Functions

- **LanguageException** ()
- **LanguageException** (String s)

Static Package Attributes

- static final long serialVersionUID = PROJECTUID + LANGUAGEEXCEPTION_SUBID

7.19.1 Detailed Description

Language(p. 61) Exception.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition at line 18 of file LanguageException.java.

7.19.2 Constructor & Destructor Documentation

7.19.2.1 lang::LanguageException::LanguageException () [inline]

Constructor

Definition at line 28 of file LanguageException.java.

7.19.2.2 lang::LanguageException::LanguageException (String s) [inline]

Constructor

Parameters:

s message for the exception

See also:

Exception(String)

Definition at line 37 of file LanguageException.java.

7.19.3 Member Data Documentation

7.19.3.1 `final long lang::LanguageException::serialVersionUID = PROJECTUID + LANGUAGEEXECPTION_SUBID` [static, package]

Object identifier

Definition at line 23 of file `LanguageException.java`.

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

- `workspace/S-TGDH_god/lang/LanguageException.java`

7.20 console::Launcher Class Reference

Launcher(p. 65) for console mode of the simulator.

Inheritance diagram for console::Launcher::

Public Member Functions

- **Launcher** ()
- void **launch** (String[] args)

Static Public Member Functions

- static void **main** (String[] args)

Static Public Attributes

- static int **MODE**
- static final int **DEBUG** = 1
- static final int **RUN** = 2

7.20.1 Detailed Description

Launcher(p. 65) for console mode of the simulator.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition at line 22 of file Launcher.java.

7.20.2 Constructor & Destructor Documentation

7.20.2.1 console::Launcher::Launcher () [inline]

Launcher(p. 65) Constructor

Definition at line 42 of file Launcher.java.

7.20.3 Member Function Documentation

7.20.3.1 void console::Launcher::launch (String[] *args*) [inline]

Classical entry point: start the launcher

Implements **manager::ILauncher** (p. 50).

Definition at line 49 of file Launcher.java.

7.20.3.2 static void console::Launcher::main (String[] *args*) [inline, static]

Common test function

Parameters:

args

Definition at line 78 of file Launcher.java.

7.20.4 Member Data Documentation

7.20.4.1 final int console::Launcher::DEBUG = 1 [static]

Display full debug messages

Definition at line 32 of file Launcher.java.

7.20.4.2 int console::Launcher::MODE [static]

Describes the mode of the launcher

Definition at line 27 of file Launcher.java.

7.20.4.3 final int console::Launcher::RUN = 2 [static]

Display only simulation messages

Definition at line 37 of file Launcher.java.

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

- workspace/S-TGDH_god/console/**Launcher.java**

7.21 manager::Launcher Class Reference

Launcher(p. 67) of the simulator.

Public Member Functions

- **Launcher** ()

Static Public Member Functions

- static void **main** (String[] args)

Static Public Attributes

- static int **MODE**
- static final int **DEBUG** = 1
- static final int **RUN** = 2

7.21.1 Detailed Description

Launcher(p. 67) of the simulator.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition at line 20 of file Launcher.java.

7.21.2 Constructor & Destructor Documentation

7.21.2.1 manager::Launcher::Launcher () [inline]

Launcher(p. 67) Constructor

Definition at line 40 of file Launcher.java.

7.21.3 Member Function Documentation

7.21.3.1 static void manager::Launcher::main (String[] args) [inline, static]

Start the launcher

Parameters:

args the eventual parameters

Note:

for help, enter help as the single parameter for the application (for instance, "java -jar //program_name// help")

Definition at line 49 of file Launcher.java.

7.21.4 Member Data Documentation

7.21.4.1 final int manager::Launcher::DEBUG = 1 [static]

Display full debug messages

Definition at line 30 of file Launcher.java.

7.21.4.2 int manager::Launcher::MODE [static]

Describes the mode of the launcher

Definition at line 25 of file Launcher.java.

7.21.4.3 final int manager::Launcher::RUN = 2 [static]

Display only simulation messages

Definition at line 35 of file Launcher.java.

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

- workspace/S-TGDH_god/manager/**Launcher.java**

7.22 ui::Launcher Class Reference

Launcher(p. 69) for graphical mode of the simulator.

Inheritance diagram for ui::Launcher::

Public Member Functions

- void **launch** (String[] args)

Static Public Member Functions

- static void **main** (String[] args)

7.22.1 Detailed Description

Launcher(p. 69) for graphical mode of the simulator.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition at line 19 of file Launcher.java.

7.22.2 Member Function Documentation

7.22.2.1 void ui::Launcher::launch (String[] args) [inline]

Classical entry point: start the launcher

Implements **manager::ILauncher** (p. 50).

Definition at line 24 of file Launcher.java.

7.22.2.2 static void ui::Launcher::main (String[] args) [inline, static]

Common test function

Parameters:

args

Definition at line 33 of file Launcher.java.

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

- workspace/S-TGDH_god/ui/**Launcher.java**

7.23 manager::Log Class Reference

Logging class.

Static Public Member Functions

- static void **init** ()
- static void **init** (int mode)
- static void **flush** ()
- static void **createLog** (int logMode, **INode** node, String message)
- static void **LogSPY** (String srcID, String destID, String message, String header, String sign)
- static void **newBloc** (**INode** node)
- static void **addHandler** (String id, JEditorPane f)
- static void **setFileHandle** (String fileName) throws IOException
- static void **terminate** ()

Static Public Attributes

- static final int **LOG_MANAGER** = 0
- static final int **LOG_MAIN** = 1
- static final int **LOG_INIT_GROUP** = 2
- static final int **LOG_SECURITY_ERROR** = 4
- static final int **LOG_CRYPTO** = 3
- static final int **LOG_SECURITY_INVARIANT** = 5
- static final int **LOG_START_GROUP** = 6
- static final int **GROUP_MANAGEMENT** = 7
- static final int **GROUP_INFO** = 8
- static final int **GROUP_UPDATE** = 9
- static final int **GROUP_ERROR** = 10
- static final String **SPY** = "SPY"
- static final int **RECORD_VERBOSE** = 0
- static final int **RECORD_SILENT** = 1

7.23.1 Detailed Description

Logging class.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition at line 25 of file Log.java.

7.23.2 Member Function Documentation

7.23.2.1 static void manager::Log::addHandler (String *id*, JEditorPane *f*) [inline, static]

Add a logging handle between a node and a panel.

Parameters:

id the node identifier

f the handler on the panel to associate to the node

Definition at line 282 of file Log.java.

7.23.2.2 static void manager::Log::createLog (int *logMode*, INode *node*, String *message*) [inline, static]

Create a log message.

Parameters:

logMode logging header

node targeted node identifier

message message to log

Definition at line 174 of file Log.java.

7.23.2.3 static void manager::Log::flush () [inline, static]

Force the log messages to be displayed.

Note:

Calls to this method is required only if the fileLogging mode is on

Definition at line 161 of file Log.java.

7.23.2.4 static void manager::Log::init (int *mode*) [inline, static]

Initialize the logging class.

Parameters:

mode record depth

See also:

`RECORD_SILENT`(p. 74)

`RECORD_VERBOSE`(p. 74)

Definition at line 135 of file Log.java.

7.23.2.5 static void manager::Log::init () [inline, static]

Initialize the logging class with the default record depth

Definition at line 125 of file Log.java.

7.23.2.6 static void manager::Log::LogSPY (String *srcID*, String *destID*, String *message*, String *header*, String *sign*) [inline, static]

Log(p. 70) information for a spy (network eavesdrooper).

Parameters:

srcID source identifier
destID destinator identifier
message message to log
header message header
sign message signature

Definition at line 250 of file Log.java.

7.23.2.7 static void manager::Log::newBloc (INode *node*) [inline, static]

Create a new comment bloc.

Parameters:

node node for which the new bloc have to be created

Definition at line 273 of file Log.java.

7.23.2.8 static void manager::Log::setFileHandle (String *fileName*) throws IOException [inline, static]

Active the fileLogging mode.

Parameters:

fileName file in which log must be recored

Exceptions:

IOException an IO Exception occured during the handling operation

Definition at line 293 of file Log.java.

7.23.2.9 static void manager::Log::terminate () [inline, static]

Terminate all the logging operations.

See also:

Supervisor::terminate()

Definition at line 303 of file Log.java.

7.23.3 Member Data Documentation

7.23.3.1 final int manager::Log::GROUP_ERROR = 10 [static]

An error occured in the group

Definition at line 84 of file Log.java.

7.23.3.2 final int manager::Log::GROUP_INFO = 8 [static]

Additional information about the group

Definition at line 74 of file Log.java.

7.23.3.3 final int manager::Log::GROUP_MANAGEMENT = 7 [static]

Group management information

Definition at line 69 of file Log.java.

7.23.3.4 final int manager::Log::GROUP_UPDATE = 9 [static]

A group update occurred

Definition at line 79 of file Log.java.

7.23.3.5 final int manager::Log::LOG_CRYPT = 3 [static]

Crypto messages

Definition at line 54 of file Log.java.

7.23.3.6 final int manager::Log::LOG_INIT_GROUP = 2 [static]

Messages sent for the group initiation (keys exchanges)

Definition at line 44 of file Log.java.

7.23.3.7 final int manager::Log::LOG_MAIN = 1 [static]

Unspecified messages

Definition at line 39 of file Log.java.

7.23.3.8 final int manager::Log::LOG_MANAGER = 0 [static]

Messages sent by the supervisor

Definition at line 34 of file Log.java.

7.23.3.9 final int manager::Log::LOG_SECURITY_ERROR = 4 [static]

Security error : either an error occurred or a bad node is present !

Definition at line 49 of file Log.java.

7.23.3.10 final int manager::Log::LOG_SECURITY_INVARIANT = 5 [static]

Errors which should not be present, if the security invariants were respected

Definition at line 59 of file Log.java.

7.23.3.11 final int manager::Log::LOG_START_GROUP = 6 [static]

The group is started

Definition at line 64 of file Log.java.

7.23.3.12 final int manager::Log::RECORD_SILENT = 1 [static]

The logging instance will record the minimal number of messages

Note:

this mode may be relevant for the filelogging mode

Definition at line 120 of file Log.java.

7.23.3.13 final int manager::Log::RECORD_VERBOSE = 0 [static]

The logging instance will record all the messages

Definition at line 114 of file Log.java.

7.23.3.14 final String manager::Log::SPY = "SPY" [static]

Additional identifier: eavesdrooper, for logging operations

Definition at line 104 of file Log.java.

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

- workspace/S-TGDH_god/manager/**Log.java**

7.24 ui::Manager Class Reference

Core window of the simulator graphical mode.

Inheritance diagram for ui::Manager::

Public Member Functions

- **Manager** (**Supervisor** *s*)
- void **removeConfirmed** (String *id*)
- void **removeCanceled** (String *id*)
- void **concludeFalse** ()
- void **concludeTrue** (String *initiator*, List< String > *listID*, int *grpPass*)
- String **addNode** ()
- void **terminate** ()

7.24.1 Detailed Description

Core window of the simulator graphical mode.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition at line 46 of file Manager.java.

7.24.2 Constructor & Destructor Documentation

7.24.2.1 ui::Manager::Manager (**Supervisor** *s*) [inline]

Create the core window of the simulator graphical mode

See also:

Manager::initObjects()
Manager::addActions()

Parameters:

s handler on the supervisor object

See also:

manager.Supervisor(p. 97)

Definition at line 171 of file Manager.java.

7.24.3 Member Function Documentation

7.24.3.1 String ui::Manager::addNode () [inline]

Add a node to the network

Returns:

String the new node identifier

Definition at line 481 of file Manager.java.

7.24.3.2 void ui::Manager::concludeFalse () [inline]

CreateGroup(p. 35) window message aborting the group creation operation

Definition at line 438 of file Manager.java.

7.24.3.3 void ui::Manager::concludeTrue (String *initiator*, List< String > *listID*, int *grpPass*) [inline]

CreateGroup(p. 35) window message confirming the group creation operation

Parameters:

initiator initiator identifier

listID group list (before confirmation)

grpPass weak group password, to assure key exchange validity

Definition at line 448 of file Manager.java.

7.24.3.4 void ui::Manager::removeCanceled (String *id*) [inline]

The removal process for a node failed

Parameters:

id identifier of the node who initiates the remove operation

See also:

[ui.Interface::removeErrorInfo\(String\)](#)(p. 56)

Definition at line 430 of file Manager.java.

7.24.3.5 void ui::Manager::removeConfirmed (String *id*) [inline]

The removal process for a node succeeded

Parameters:

id identifier of the node who initiates the remove operation

See also:

[ui.Interface::removeConfirmInfo\(String\)](#)(p. 56)

Definition at line 420 of file Manager.java.

7.24.3.6 void ui::Manager::terminate () [inline]

Reinitialize the graphical windows states

See also:

manager.Supervisor::terminate()
Manager::addActions()

Definition at line 524 of file Manager.java.

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

- workspace/S-TGDH_god/ui/**Manager.java**

7.25 manager::messages::Message Class Reference

Unicast message stored in the network boxes.

Inheritance diagram for manager::messages::Message::

Public Member Functions

- **Message** (INode dest, String h, String m, String sign, String sID)

Public Attributes

- INode **nodeDest**
- String **header**
- String **message**
- String **srcID**
- String **signature**
- long **RTT**

7.25.1 Detailed Description

Unicast message stored in the network boxes.

See also:

manager.Supervisor::broadcastGroup(INode, String, String, String)

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 20 of file Message.java.

7.25.2 Constructor & Destructor Documentation

7.25.2.1 manager::messages::Message::Message (INode *dest*, String *h*, String *m*, String *sign*, String *sID*) [inline]

Create an unicast message

Parameters:

dest the node to reach

h the message header

See also:

`manager.SupervisorMessage`(p. 98)

Parameters:

m the message

sign the message signature

sID the identifier of the source

Definition at line 61 of file Message.java.

7.25.3 Member Data Documentation

7.25.3.1 String manager::messages::Message::header

message header

See also:

`manager.SupervisorMessage`(p. 98)

Definition at line 31 of file Message.java.

7.25.3.2 String manager::messages::Message::message

`Message`(p. 78) payload

Definition at line 36 of file Message.java.

7.25.3.3 INode manager::messages::Message::nodeDest

Node to reach

Definition at line 25 of file Message.java.

7.25.3.4 long manager::messages::Message::RTT

`Message`(p. 78) RTT

Definition at line 51 of file Message.java.

7.25.3.5 String manager::messages::Message::signature

`Message`(p. 78) signature

Definition at line 46 of file Message.java.

7.25.3.6 String manager::messages::Message::srcID

Source identifier

Definition at line 41 of file Message.java.

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

- `workspace/S-TGDH_god/manager/messages/Message.java`

7.26 manager::messages::MulticastMessage Class Reference

Multicast message stored in the network boxes.

Inheritance diagram for manager::messages::MulticastMessage::

Public Member Functions

- **MulticastMessage** (List< String > grp, String h, String m, String sign, String sID)

Public Attributes

- List< String > grpList

7.26.1 Detailed Description

Multicast message stored in the network boxes.

See also:

manager.Supervisor::broadcastGroup(INode, String, String, String)

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 20 of file MulticastMessage.java.

7.26.2 Constructor & Destructor Documentation

7.26.2.1 manager::messages::MulticastMessage::MulticastMessage (List< String > grp, String h, String m, String sign, String sID) [inline]

Create a multicast message

Parameters:

grp the nodes to reach

h the message header

See also:

manager.SupervisorMessage(p. 98)

Parameters:

m the message

sign the message signature

sID the identifier of the source

Definition at line 35 of file MulticastMessage.java.

7.26.3 Member Data Documentation

7.26.3.1 List<String> manager::messages::MulticastMessage::grpList

The list of nodes concerned by the message

Definition at line 25 of file MulticastMessage.java.

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

- workspace/S-TGDH_god/manager/messages/**MulticastMessage.java**

7.27 node::Node Class Reference

Modelisation of a node of the network.

Inheritance diagram for node::Node:

Public Member Functions

- **Node** ()
- **Node** (**Supervisor** superVis)
- synchronized void **init** ()
- **Tree** **getTree** ()
- String **getID** ()
- synchronized void **receive** (String header, String message, String signature, String nodeSrc-ID)
- List< **NodeMessage** > **getBox** ()
- void **setGroupPassword** (int password)
- void **initGroup** (List< String > nodesListID)
- void **purge** ()
- Override void **run** ()
- Override void **start** ()
- void **terminate** ()
- synchronized String **addNode2Group** (**INode** n, long GK)
- synchronized int **canAddNewNode** ()
- boolean **updateKeyBeforeAdd** (String node2addID)
- synchronized String **joinGroup** (**INode** sponsorNode)
- List< String > **getGroupIDs** ()
- long **get_RSA_pub** ()
- boolean **removeFromGroup** ()
- int **getGroupState** ()
- long **getGroupKey** ()
- synchronized String **getRemainingMessagesView** ()

Protected Member Functions

- void **setGroupState** (int state)

Protected Attributes

- long **RSA_priv**
- long **RSA_public**
- Hashtable< String, Long > **RSA_keys_table**
- **Supervisor** **_supervisor**
- String **ID**
- **InitNode** **init_behav**
- **GroupNode** **group_behav**

7.27.1 Detailed Description

Modelisation of a node of the network.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/22

Definition at line 27 of file Node.java.

7.27.2 Constructor & Destructor Documentation

7.27.2.1 `node::Node::Node ()` [inline]

Constructor, for virtual nodes

Definition at line 78 of file Node.java.

7.27.2.2 `node::Node::Node (Supervisor superVis)` [inline]

Constructor

Parameters:

superVis handle on the network simulator

Definition at line 86 of file Node.java.

7.27.3 Member Function Documentation

7.27.3.1 `synchronized String node::Node::addNode2Group (INode n, long GK)` [inline]

This method is called by the node to be added to ask the sponsor to add it to the group

The standard behavior is : + sponsor.updateKeyBeforeAdd + newNode.joinGroup() + newNode.initFromSponsor() <- retrieve the current information of the group, after key updates ! + sponsor.addNode2Group() <- tell the other members of the group that a node is to be added

See also:

`GroupNode::addNodeSponsor(INode, long)`(p. 43)
`joinGroup(INode)`(p. 86)

Implements `node::INode` (p. 55).

Definition at line 496 of file Node.java.

7.27.3.2 `synchronized int node::Node::canAddNewNode ()` [inline]

Interface the group behavior

See also:

`IGroupNode::canAddNewNode()`(p. 47)

Implements `node::IGroupNode` (p. 47).

Definition at line 505 of file `Node.java`.

7.27.3.3 `long node::Node::get_RSA_pub ()` [inline]

Return the RSA public key of the node

Implements `node::INode` (p. 55).

Definition at line 583 of file `Node.java`.

7.27.3.4 `List<NodeMessage> node::Node::getBox ()` [inline]

Return the list of messages of the nodes (required for nodes synchronisations)

Implements `node::INode` (p. 55).

Definition at line 166 of file `Node.java`.

7.27.3.5 `List<String> node::Node::getGroupIDs ()` [inline]

Interface the group behavior

See also:

`GroupNode::getNodesList()`(p. 44)

Implements `node::INode` (p. 55).

Definition at line 572 of file `Node.java`.

7.27.3.6 `long node::Node::getGroupKey ()` [inline]

Method for statistics Return the group key of the current node

Implements `node::INode` (p. 55).

Definition at line 618 of file `Node.java`.

7.27.3.7 `int node::Node::getGroupState ()` [inline]

Return the node state

See also:

`NodeConstants`(p. 92)

Implements `node::INode` (p. 55).

Definition at line 600 of file `Node.java`.

7.27.3.8 String node::Node::getID () [inline]

return the ID of the node

Implements **node::INode** (p. 55).

Definition at line 123 of file Node.java.

7.27.3.9 synchronized String node::Node::getRemainingMessagesView () [inline]

Method for statistics Display the remaining messages of the box

Implements **node::INode** (p. 55).

Definition at line 628 of file Node.java.

7.27.3.10 Tree node::Node::getTree () [inline]

Return the node view of the cryptographic tree

Implements **node::INode** (p. 55).

Definition at line 114 of file Node.java.

7.27.3.11 synchronized void node::Node::init () [inline]

initialize the node state

Definition at line 104 of file Node.java.

7.27.3.12 void node::Node::initGroup (List< String > nodesListID) [inline]

Method to be called once on the node which acts as the initiator

Parameters:

nodesListID all the node supposed to be in the futur group

Implements **node::IInitNode** (p. 49).

Definition at line 374 of file Node.java.

7.27.3.13 synchronized String node::Node::joinGroup (INode sponsorNode) [inline]

The node start the procedure to join the group, try to do it with the sponsor node in parameter

The new node adds itself to the group diffusion list, but in reality, it can be also true (but if the group doesn't him to join, then it won't be able to decrypt the messages)

To operate like in real life, the method

See also:

updateKeyBeforeAdd(String)(p. 88) is supposed to be called just before this method.

The standard behavior is : + sponsor.updateKeyBeforeAdd + newNode.joinGroup() + newNode.initFromSponsor() <- retrieve the current information of the group, after key updates ! + sponsor.addNode2Group() <- tell the other members of the group that a node is to be added

Parameters:

sponsorNode the node supposed to act as a sponsor

Returns:

String null if everything goes well, an error message otherwise

Implements **node::INode** (p. 55).

Definition at line 545 of file Node.java.

7.27.3.14 void node::Node::purge () [inline]

remove all the unneeded messages All the next messages have a RTT difference $> 2 * \text{propagation_delay}$... unneeded message (and check() $< \text{RTT}$!)

Definition at line 394 of file Node.java.

7.27.3.15 synchronized void node::Node::receive (String header, String message, String signature, String nodeSrcID) [inline]

upon receiving a message, check it's identity (if possible) and then manage it It calls the do-Message() method

Parameters:

header

message

signature

nodeSrcID

Implements **node::INode** (p. 55).

Definition at line 157 of file Node.java.

7.27.3.16 boolean node::Node::removeFromGroup () [inline]

The node wants to leave the group

See also:

GroupNode::removeFromGroup()(p. 45)

Implements **node::IGroupNode** (p. 47).

Definition at line 591 of file Node.java.

7.27.3.17 Override void node::Node::run () [inline]

Start the thread which model the node

Definition at line 425 of file Node.java.

7.27.3.18 void node::Node::setGroupPassword (int *password*) [inline]

Defined the group password. This method is supposed to be called one's and represents the ASOKAN-GINZBOORG key initiation protocol

Implements **node::INode** (p. 55).

Definition at line 364 of file Node.java.

7.27.3.19 void node::Node::setGroupState (int *state*) [inline, protected]

Modify the group state

Parameters:

state

Definition at line 608 of file Node.java.

7.27.3.20 Override void node::Node::start () [inline]

Thread function override

Definition at line 465 of file Node.java.

7.27.3.21 void node::Node::terminate () [inline]

ask the node to start the terminate procedure The node will stop itself only if it is not in a current operation such group creation

See also:

run()(p. 87)

Implements **node::INode** (p. 55).

Definition at line 477 of file Node.java.

7.27.3.22 boolean node::Node::updateKeyBeforeAdd (String *node2addID*) [inline]

The node which acts as a sponsor needs to build a new pair of key Note that's this method is supposed to be implemented in reality as a discussion between the new node and the one that will act as its sponsor

Parameters:

node2addID the id of the node to be added (needed for log messages)

See also:

GroupNode::updateKeyBeforeAdd(String)(p. 46)

Implements **node::IGroupNode** (p. 48).

Definition at line 518 of file Node.java.

7.27.4 Member Data Documentation

7.27.4.1 Supervisor node::Node::_supervisor [protected]

Handle on the network simulator

Definition at line 57 of file Node.java.

7.27.4.2 GroupNode node::Node::group_behav [protected]

Handle on the group behavior

Definition at line 73 of file Node.java.

7.27.4.3 String node::Node::ID [protected]

node's identifier

Definition at line 62 of file Node.java.

7.27.4.4 InitNode node::Node::init_behav [protected]

Handle on the initiation behavior

Definition at line 68 of file Node.java.

7.27.4.5 Hashtable<String, Long> node::Node::RSA_keys_table [protected]

RSA keys of the different nodes

Definition at line 45 of file Node.java.

7.27.4.6 long node::Node::RSA_priv [protected]

RSA private key

Definition at line 33 of file Node.java.

7.27.4.7 long node::Node::RSA_public [protected]

RSA public key, for authentication

Definition at line 38 of file Node.java.

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

- workspace/S-TGDH_god/node/Node.java

7.28 node::NodeBehavior Class Reference

Model the standard behavior of a node.

Inheritance diagram for node::NodeBehavior::

Public Member Functions

- **NodeBehavior** (**Node** *m*)
- abstract void **init** ()

Protected Attributes

- **Node** *main*

7.28.1 Detailed Description

Model the standard behavior of a node.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 16 of file NodeBehavior.java.

7.28.2 Constructor & Destructor Documentation

7.28.2.1 node::NodeBehavior::NodeBehavior (**Node** *m*) [inline]

Constructor

Parameters:

m Handle on the associated node

Definition at line 27 of file NodeBehavior.java.

7.28.3 Member Function Documentation

7.28.3.1 abstract void node::NodeBehavior::init () [pure virtual]

Initialize the behavior state

Implemented in **node::GroupNode** (p. 44), and **node::InitNode** (p. 53).

7.28.4 Member Data Documentation

7.28.4.1 Node node::NodeBehavior::main [protected]

node associated to the specific behavior

Definition at line 21 of file NodeBehavior.java.

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

- workspace/S-TGDH_god/node/**NodeBehavior.java**

7.29 node::NodeConstants Interface Reference

Interface for the nodes: contains standard constants.

Inheritance diagram for node::NodeConstants:

Static Public Attributes

- static final int **NOTHING** = -1
- static final int **GROUP_INITIATOR** = 0
- static final int **GROUP_CREATING** = 1
- static final int **GROUP_MEMBER** = 2
- static final int **GROUP_LEAVING** = 3
- static final int **WAITING** = 4
- static final int **NODE_SLEEP_TIME** = 20

7.29.1 Detailed Description

Interface for the nodes: contains standard constants.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 18 of file NodeConstants.java.

7.29.2 Member Data Documentation

7.29.2.1 final int node::NodeConstants::GROUP_CREATING = 1 [static]

The node is currently in the group creation process

Definition at line 33 of file NodeConstants.java.

7.29.2.2 final int node::NodeConstants::GROUP_INITIATOR = 0 [static]

The node is currently the group initiator

Definition at line 28 of file NodeConstants.java.

7.29.2.3 final int node::NodeConstants::GROUP_LEAVING = 3 [static]

The node is in the leaving process

Definition at line 43 of file NodeConstants.java.

7.29.2.4 final int node::NodeConstants::GROUP_MEMBER = 2 [static]

The node is a member of the group

Definition at line 38 of file NodeConstants.java.

7.29.2.5 final int node::NodeConstants::NODE_SLEEP_TIME = 20 [static]

Waiting time between each messages check

Definition at line 54 of file NodeConstants.java.

7.29.2.6 final int node::NodeConstants::NOTHING = -1 [static]

The node is in the default state

Definition at line 23 of file NodeConstants.java.

7.29.2.7 final int node::NodeConstants::WAITING = 4 [static]

The node is in a synchronization process. It Collects messages but do not manage them (later).

Definition at line 49 of file NodeConstants.java.

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

- workspace/S-TGDH_god/node/**NodeConstants.java**

7.30 ui::NodeFrame Class Reference

Window use to retrieve information on a node of the network.

Inheritance diagram for ui::NodeFrame::

Public Member Functions

- **NodeFrame** (String *title*, **Node** *n*, JScrollPane *jsb*)
- void **initActionFrame** ()

7.30.1 Detailed Description

Window use to retrieve information on a node of the network.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition at line 35 of file NodeFrame.java.

7.30.2 Constructor & Destructor Documentation

7.30.2.1 ui::NodeFrame::NodeFrame (String *title*, Node *n*, JScrollPane *jsb*) [inline]

Parameters:

title title associated to the window

n handler on the associated node

jsb swing component associated to the Log class

See also:

manager.Log(p. 70)

Definition at line 72 of file NodeFrame.java.

7.30.3 Member Function Documentation

7.30.3.1 void ui::NodeFrame::initActionFrame () [inline]

Initialisaiton of the actionFrame component: layout and sub components

Definition at line 96 of file NodeFrame.java.

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

- workspace/S-TGDH_god/ui/**NodeFrame.java**

7.31 node::messages::NodeMessage Class Reference

Modelize a message in the node stack.

Public Member Functions

- String **getHeader** ()
- String **getMessage** ()
- long **getRTT** ()
- void **clearRTT** ()
- String **getSignature** ()
- String **getSrcID** ()
- **NodeMessage** (String *h*, String *m*, String *sign*, String *sID*)
- String **toString** ()

7.31.1 Detailed Description

Modelize a message in the node stack.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition at line 17 of file NodeMessage.java.

7.31.2 Constructor & Destructor Documentation

7.31.2.1 node::messages::NodeMessage::NodeMessage (String *h*, String *m*, String *sign*, String *sID*) [inline]

Create a node message (for the stack)

Parameters:

h the message header

See also:

[manager.SupervisorMessage](#)(p. 98)

Parameters:

m the message

sign the message signature

sID the identifier of the source

Definition at line 96 of file NodeMessage.java.

7.31.3 Member Function Documentation

7.31.3.1 void node::messages::NodeMessage::clearRTT () [inline]

Reset the RTT

Note:

As the RTT is used to check whether a message is still valid, reinitialisation may be required, though it is not the way it is done in reality (additional timers)

Definition at line 71 of file NodeMessage.java.

7.31.3.2 String node::messages::NodeMessage::getHeader () [inline]

Return the message header

Definition at line 47 of file NodeMessage.java.

7.31.3.3 String node::messages::NodeMessage::getMessage () [inline]

Return the message payload

Definition at line 54 of file NodeMessage.java.

7.31.3.4 long node::messages::NodeMessage::getRTT () [inline]

Return the message RTT

Definition at line 61 of file NodeMessage.java.

7.31.3.5 String node::messages::NodeMessage::getSignature () [inline]

Return the message signature

Definition at line 78 of file NodeMessage.java.

7.31.3.6 String node::messages::NodeMessage::getSrcID () [inline]

Return the identifier of the source

Definition at line 85 of file NodeMessage.java.

7.31.3.7 String node::messages::NodeMessage::toString () [inline]

Convert the message to a readable String

Definition at line 107 of file NodeMessage.java.

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

- workspace/S-TGDH_god/node/messages/NodeMessage.java

7.32 manager::Supervisor Class Reference

Network modelization.

7.32.1 Detailed Description

Network modelization.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition at line 24 of file Supervisor.java.

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

- workspace/S-TGDH_god/manager/**Supervisor.java**

7.33 manager::SupervisorMessage Interface Reference

Messages modelisation for the interaction between the nodes (and the network).

Static Public Attributes

- static final String **INCORRECT_DEST** = "-1"
- static final String **SUPERVISOR_ERROR** = "-2"
- static final String **INIT_GROUP** = "1"
- static final String **ANSWER_INIT_GROUP** = "2"
- static final String **FINISH_INIT_GROUP** = "3"
- static final String **CANCEL_INIT_GROUP** = "4"
- static final String **GROUP_INIT_GKPERSO** = "5"
- static final String **GROUP_INIT_KPERSO** = "6"
- static final String **GROUP_ADD_OPERATION** = "8"
- static final String **GROUP_REMOVE_SPONSOR** = "9"
- static final String **GROUP_REMOVE_OPERATION** = "10"
- static final String **GROUP_REMOVE_SPONSOR_CONFIRM** = "11"

7.33.1 Detailed Description

Messages modelisation for the interaction between the nodes (and the network).

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition at line 18 of file SupervisorMessage.java.

7.33.2 Member Data Documentation

7.33.2.1 final String manager::SupervisorMessage::ANSWER_INIT_GROUP = "2" [static]

Messages for the second step of the initiation

Note:

Part of the group formation (IKA) messages

Definition at line 44 of file SupervisorMessage.java.

7.33.2.2 `final String manager::SupervisorMessage::CANCEL_INIT_GROUP = "4" [static]`

Messages for the forth step of the initiation

Note:

Part of the group formation (IKA) messages

Definition at line 56 of file SupervisorMessage.java.

7.33.2.3 `final String manager::SupervisorMessage::FINISH_INIT_GROUP = "3" [static]`

Messages for the thrid step of the initiation

Note:

Part of the group formation (IKA) messages

Definition at line 50 of file SupervisorMessage.java.

7.33.2.4 `final String manager::SupervisorMessage::GROUP_ADD_OPERATION = "8" [static]`

Messages for the adding opertion, from the sponsor

Note:

Part of the group management (AKA) messages

Definition at line 77 of file SupervisorMessage.java.

7.33.2.5 `final String manager::SupervisorMessage::GROUP_INIT_GKPERSO = "5" [static]`

Messages for the GK exchange

Note:

Part of the group management (AKA) messages

Definition at line 64 of file SupervisorMessage.java.

7.33.2.6 `final String manager::SupervisorMessage::GROUP_INIT_KPERSO = "6" [static]`

Messages for the K exchange

Note:

Part of the group management (AKA) messages

Definition at line 70 of file SupervisorMessage.java.

7.33.2.7 final String manager::SupervisorMessage::GROUP_REMOVE_OPERATION = "10" [static]

Messages for the GK remove information from a node

Note:

Part of the group management (AKA) messages

Definition at line 89 of file SupervisorMessage.java.

7.33.2.8 final String manager::SupervisorMessage::GROUP_REMOVE_SPONSOR = "9" [static]

Messages for the remove operation from the sponsor

Note:

Part of the group management (AKA) messages

Definition at line 83 of file SupervisorMessage.java.

7.33.2.9 final String manager::SupervisorMessage::GROUP_REMOVE_SPONSOR_CONFIRM = "11" [static]

Messages for the remove operation from the sponsor, confirmation to the asking node

Note:

Part of the group management (AKA) messages

Definition at line 96 of file SupervisorMessage.java.

7.33.2.10 final String manager::SupervisorMessage::INCORRECT_DEST = "-1" [static]

The destination nodes do not exist

Note:

Part of the **Supervisor**(p. 97) error messages

Definition at line 25 of file SupervisorMessage.java.

7.33.2.11 final String manager::SupervisorMessage::INIT_GROUP = "1" [static]

Messages for the first step of the initiation

Note:

Part of the group formation (IKA) messages

Definition at line 38 of file SupervisorMessage.java.

```
7.33.2.12 final String manager::SupervisorMessage::SUPERVISOR_ERROR =  
        "-2" [static]
```

Undefined supervisor error

Note:

Part of the **Supervisor**(p. 97) error messages

Definition at line 31 of file SupervisorMessage.java.

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

- workspace/S-TGDH_god/manager/**SupervisorMessage.java**

7.34 tools::Tree Class Reference

Represents the cryptographic tree of the (S-)TGDH algorithm.

Public Member Functions

- **Tree** ()
- **TreeNode** **getRoot** ()
- void **addEnd** (**TreeNode** treeNode)
- boolean **replaceSubTree** (**TreeNode** oldNode, **TreeNode** n)
- int **addNeighbour** (**TreeNode** neighbour, **TreeNode** n)
- boolean **remove** (String nodeID)
- **TreeNode** **getTreeNode** (String id)
- String **toString** (**TreeNode** node)
- String **toString** ()
- String **display** ()
- List< String > **getNodesList** ()
- int **getMinDepth** ()

Static Public Member Functions

- static **Tree** **fetch** (String s_tree)
- static void **main** (String[] args)

Static Public Attributes

- static long **NOTaNODE** = -11

7.34.1 Detailed Description

Represents the cryptographic tree of the (S-)TGDH algorithm.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition at line 19 of file Tree.java.

7.34.2 Constructor & Destructor Documentation

7.34.2.1 tools::Tree::Tree () [inline]

Constructor

Definition at line 34 of file Tree.java.

7.34.3 Member Function Documentation

7.34.3.1 void tools::Tree::addEnd (TreeNode *treeNode*) [inline]

Add a node to the end of the tree and rename the virtual nodes

Parameters:

treeNode node to add

Definition at line 75 of file Tree.java.

7.34.3.2 int tools::Tree::addNeighbour (TreeNode *neighbourh*, TreeNode *n*) [inline]

Add the node *n* using the sponsor node *neighbour* and rename the virtual nodes If the parent node is virtual, *n* = parent. Otherwise, create a virtual node and its childs ars *n* & *neighbour*

Parameters:

neighbourh

n nodeToAdd

Returns:

int -1 if error, 1 if replace parent, 2 if create a virtual node

Definition at line 181 of file Tree.java.

7.34.3.3 String tools::Tree::display () [inline]

Display the tree in a convenient way

Returns:

the String to be displayed

Definition at line 458 of file Tree.java.

7.34.3.4 static Tree tools::Tree::fetch (String *s_tree*) [inline, static]

Build a tree associated to it's String (

See also:

[toString\(\)](#)(p. 105))

Parameters:

s_tree the message which represents the tree

Returns:

the tree

the recursive structure is (node, [child1], [child2]) and node structure's is {} or {;}

Definition at line 417 of file Tree.java.

7.34.3.5 `int tools::Tree::getMinDepth ()` [inline]

return the minimal depth of the tree (ie the height of the smaller branch)

Returns:

the minimal depth

Definition at line 483 of file Tree.java.

7.34.3.6 `List<String> tools::Tree::getNodesList ()` [inline]

Return all the id of the nodes in the tree

Returns:

a list which contains all the id

Definition at line 473 of file Tree.java.

7.34.3.7 `TreeNode tools::Tree::getRoot ()` [inline]

Return the root of the tree

Returns:

Definition at line 43 of file Tree.java.

7.34.3.8 `TreeNode tools::Tree::getTreeNode (String id)` [inline]

Return the node of the tree which has the id in paramaters

Parameters:

id the node ID

Returns:

the associated node or null if it doesn't exist

Definition at line 368 of file Tree.java.

7.34.3.9 `static void tools::Tree::main (String[] args)` [inline, static]

Common test function

Parameters:

args no parameter is required

Definition at line 491 of file Tree.java.

7.34.3.10 boolean tools::Tree::remove (String *nodeID*) [inline]

Remove the node in parameter from the group and rename the virtual nodes

Parameters:

nodeID node to remove

Returns:

true if node removed. false otherwise (does it exists ?)

Definition at line 245 of file Tree.java.

7.34.3.11 boolean tools::Tree::replaceSubTree (TreeNode *oldNode*, TreeNode *n*) [inline]

Replace the subTree represented bu *oldNode* by the new one whose root is *n*

Parameters:

oldNode the old subtree root

n the new subtree root

Returns:

true if success, false if a real node already in the group !

Definition at line 157 of file Tree.java.

7.34.3.12 String tools::Tree::toString () [inline]

Translate the tree to a String

Returns:

the String message associated to the tree

Definition at line 399 of file Tree.java.

7.34.3.13 String tools::Tree::toString (TreeNode *node*) [inline]

Translate the tree to a String Extension of **toString()**(p. 105)

Parameters:

node the current node in the tree to translate

Returns:

the String message associated to the subtree

Definition at line 384 of file Tree.java.

7.34.4 Member Data Documentation

7.34.4.1 `long tools::Tree::NOTaNODE = -11` [static]

The requested node does not exist

Definition at line 23 of file Tree.java.

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

- workspace/S-TGDH_god/tools/**Tree.java**

7.35 tools::TreeNode Class Reference

Represents the standard element of the **Tree**(p. 102) structure.

Public Member Functions

- **TreeNode** ()
- **TreeNode** (String id, long rsa_pub)
- **TreeNode** (String id, long rsa_pub, long GK)
- void **setGK** (long gk)
- boolean **isLeftChild** ()
- **TreeNode** **findVirtual** ()
- **TreeNode** **getTreeNode** (String id)
- boolean **isLeaf** ()
- boolean **isRoot** ()
- boolean **isVirtual** ()
- List< String > **getNodesList** ()
- String **toString** ()
- String[] **display** (String decal)
- int **getDepth** ()

Static Public Member Functions

- static **TreeNode** **fetch** (String treenode)

Public Attributes

- String **ID**
- long **RSA_public_key**
- long **GK_key**
- **TreeNode** **parent**
- **TreeNode** **left_child**
- **TreeNode** **right_child**

Static Public Attributes

- static final long **INVALIDGK** = -1

Protected Member Functions

- int **getMinDepth** ()
- int **getMaxDepth** ()

Protected Attributes

- int **direction**

7.35.1 Detailed Description

Represents the standard element of the **Tree**(p. 102) structure.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition at line 19 of file `TreeNode.java`.

7.35.2 Constructor & Destructor Documentation

7.35.2.1 `tools::TreeNode::TreeNode ()` [inline]

TreeNode(p. 107) constructor

Definition at line 65 of file `TreeNode.java`.

7.35.2.2 `tools::TreeNode::TreeNode (String id, long rsa_pub)` [inline]

TreeNode(p. 107) constructor

Parameters:

id identifier of the node

rsa_pub public RSA key of the node

Definition at line 84 of file `TreeNode.java`.

7.35.2.3 `tools::TreeNode::TreeNode (String id, long rsa_pub, long GK)` [inline]

TreeNode(p. 107) constructor

Parameters:

id identifier of the node

rsa_pub public RSA key of the node

GK GK key (TGDH principle) of the node

Definition at line 94 of file `TreeNode.java`.

7.35.3 Member Function Documentation

7.35.3.1 `String [] tools::TreeNode::display (String decal)` [inline]

Perform a String representation of the tree rooted by the current node.

Parameters:

decal the space blank used at the beginning of each line

Returns:

The list of lines representing the tree rooted by the current node

Definition at line 235 of file TreeNode.java.

7.35.3.2 static TreeNode tools::TreeNode::fetch (String *treenode*) [inline, static]

Convert a String representation of the node to a **TreeNode**(p. 107).

Parameters:

treenode String representation of the node

Returns:

converted **TreeNode**(p. 107) structure

Definition at line 129 of file TreeNode.java.

7.35.3.3 TreeNode tools::TreeNode::findVirtual () [inline]

Return the first virtual node found inside the subtree. formed by the current node

Returns:

the virtual node if found, null otherwise

Definition at line 146 of file TreeNode.java.

7.35.3.4 int tools::TreeNode::getDepth () [inline]

Return the depth at which the current node is

Returns:

the depth

Definition at line 265 of file TreeNode.java.

7.35.3.5 int tools::TreeNode::getMaxDepth () [inline, protected]

Return the maximal depth of the tree

Returns:

int the maximal depth

Definition at line 286 of file TreeNode.java.

7.35.3.6 `int tools::TreeNode::getMinDepth ()` [inline, protected]

Return the minimal depth of the tree

Returns:

int the minimal depth

Definition at line 275 of file `TreeNode.java`.

7.35.3.7 `List<String> tools::TreeNode::getNodesList ()` [inline]

Retrieve the list of the non-virtual nodes in the tree.

Returns:

List<String> the list of the non-virtual nodes

Definition at line 210 of file `TreeNode.java`.

7.35.3.8 `TreeNode tools::TreeNode::getTreeNode (String id)` [inline]

Return the `TreeNode`(p. 107) associated to the `id` in parameter.

Parameters:

id identifier of the node to retrieve

Returns:

the `TreeNode`(p. 107) structure of the required node

Definition at line 163 of file `TreeNode.java`.

7.35.3.9 `boolean tools::TreeNode::isLeaf ()` [inline]

Evaluate the position of the node.

Returns:

true if the node is a leaf

Definition at line 183 of file `TreeNode.java`.

7.35.3.10 `boolean tools::TreeNode::isLeftChild ()` [inline]

Evaluate the position of the node.

Returns:

true if the node is a left member in the tree

Definition at line 119 of file `TreeNode.java`.

7.35.3.11 boolean tools::TreeNode::isRoot () [inline]

Evaluate the position of the node.

Returns:

true if the node is the root of the tree

Definition at line 192 of file TreeNode.java.

7.35.3.12 boolean tools::TreeNode::isVirtual () [inline]

Evaluate the state of the node.

Returns:

true if the node is a virtual node

Definition at line 201 of file TreeNode.java.

7.35.3.13 void tools::TreeNode::setGK (long *gk*) [inline]

Assign a GK key to the node

Parameters:

gk

Definition at line 111 of file TreeNode.java.

7.35.3.14 String tools::TreeNode::toString () [inline]

Convert the node to a String representation.

Returns:

String the String representation of the node

Definition at line 224 of file TreeNode.java.

7.35.4 Member Data Documentation**7.35.4.1** int tools::TreeNode::direction [protected]

information for the tree creation

Definition at line 44 of file TreeNode.java.

7.35.4.2 long tools::TreeNode::GK_key

node group key (public GK key)

Definition at line 39 of file TreeNode.java.

7.35.4.3 String tools::TreeNode::ID

node identifier

Definition at line 29 of file `TreeNode.java`.

7.35.4.4 final long tools::TreeNode::INVALIDGK = -1 [static]

Common identifier for invalid partial group key

Definition at line 24 of file `TreeNode.java`.

7.35.4.5 TreeNode tools::TreeNode::left_child

handle of the left child in the tree, if exists

Definition at line 54 of file `TreeNode.java`.

7.35.4.6 TreeNode tools::TreeNode::parent

Handle of the parent in the tree, if exists

Definition at line 49 of file `TreeNode.java`.

7.35.4.7 TreeNode tools::TreeNode::right_child

handle of the right child in the tree, if exists

Definition at line 59 of file `TreeNode.java`.

7.35.4.8 long tools::TreeNode::RSA_public_key

node authentication (RSA) key

Definition at line 34 of file `TreeNode.java`.

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

- `workspace/S-TGDH_god/tools/TreeNode.java`

7.36 ui::XGridBag Class Reference

The **XGridBag**(p. 113) helps to use the GridBagConstraints.

This class defines styles reusable by different components in a grid.

Inheritance diagram for ui::XGridBag::

Public Member Functions

- void **add** (Component c, **CellStyle** style, int row, int col)
- void **add** (Component c, **CellStyle** style, int row, int col, int rowHeight, int colWidth)
- **XGridBag** (Container container)

7.36.1 Detailed Description

The **XGridBag**(p. 113) helps to use the GridBagConstraints.

This class defines styles reusable by different components in a grid.

To create a layout do the following step:

- create a GridBagLayout and a **XGridBag**(p. 113).
- prepare the components size (min/prefered/max).
- create one or more style using **CellStyle.getStyle** method
- add components to the grid using the **XGridBag.add**(p. 114) method.

Example:

```
GridBagLayout layout = new GridBagLayout();
panel.setLayout(layout);
XGridBag(p. 113) grid = new XGridBag(panel);
CellStyle(p. 25) style1 = new CellStyle(p. 25)(1.0, 0.0,
GridBagConstraints.WEST, GridBagConstraints.VERTICAL, insets, 4, 0);
...
grid.add(comp1, style1, 0, 0);
grid.add(comp2, style1, 0, 1);
grid.add(comp3, style2, 1, 0);
..
```

See also:

CellStyle(p. 25)
java.awt.GridBagConstraints
java.awt.Insets

Date:

2007/06/13

Version:

1.0

Author:

//Web source//

Definition at line 70 of file XGridBag.java.

7.36.2 Constructor & Destructor Documentation

7.36.2.1 ui::XGridBag::XGridBag (Container *container*) [inline]

XGridBag(p. 113) constructor.

Parameters:

container

Definition at line 127 of file XGridBag.java.

7.36.3 Member Function Documentation

7.36.3.1 void ui::XGridBag::add (Component *c*, CellStyle *style*, int *row*, int *col*, int *rowHeight*, int *colWidth*) [inline]

Add a component to the container.

Parameters:

c component to add

style cell style

row first Y cell

col first X cell

rowHeight number of cells in a column (use GridBagConstraints.REMAINDER to fill out all rows to the end).

colWidth number of cells in a row (use GridBagConstraints.REMAINDER to fill out all columns to the end).

See also:

CellStyle(p. 25)

Definition at line 107 of file XGridBag.java.

7.36.3.2 void ui::XGridBag::add (Component *c*, CellStyle *style*, int *row*, int *col*) [inline]

Add a component to the container. The component occupy only one cell width and heigth.

Parameters:

c component to add

style cell style

row first Y cell

col first X cell

See also:

[CellStyle](#)(p. 25)

Definition at line 92 of file XGridBag.java.

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

- workspace/S-TGDH_god/ui/**XGridBag.java**

Chapter 8

javaSTGDH File Documentation

8.1 workspace/S-TGDH_god/console/Aleatory.java File Reference

Aleatory manager for the console mode of the simulator.

Namespaces

- namespace **console**

Classes

- class **console::Aleatory**
Manager for the console mode of the simulator.

8.1.1 Detailed Description

Aleatory manager for the console mode of the simulator.

Author:
Julien Thomas

Version:
1.0

Date:
2007/06/24

Definition in file **Aleatory.java**.

8.2 workspace/S-TGDH_god/console/Launcher.java File Reference

Launcher for console mode of the simulator.

Namespaces

- namespace **console**

Classes

- class **console::Launcher**
Launcher(p. 65) *for console mode of the simulator.*

8.2.1 Detailed Description

Launcher for console mode of the simulator.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition in file **Launcher.java**.

8.3 workspace/S-TGDH_god/manager/Launcher.java File Reference

Launcher of the simulator.

Namespaces

- namespace **manager**

Classes

- class **manager::Launcher**
Launcher(p. 67) *of the simulator.*

8.3.1 Detailed Description

Launcher of the simulator.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition in file **Launcher.java**.

8.4 workspace/S-TGDH_god/ui/Launcher.java File Reference

Launcher for graphical mode of the simulator.

Namespaces

- namespace **ui**

Classes

- class **ui::Launcher**
Launcher(p. 69) *for graphical mode of the simulator.*

8.4.1 Detailed Description

Launcher for graphical mode of the simulator.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition in file **Launcher.java**.

8.5 workspace/S-TGDH_god/lang/Language.java File Reference

Language manager for the simulator.

Namespaces

- namespace **lang**

Classes

- class **lang::Language**
Language(p. 61) *manager for the simulator.*

8.5.1 Detailed Description

Language manager for the simulator.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition in file **Language.java**.

8.6 workspace/S-TGDH_god/lang/Language-Exception.java File Reference

Language Exception.

Namespaces

- namespace **lang**

Classes

- class **lang::LanguageException**
Language(p. 61) *Exception*.

8.6.1 Detailed Description

Language Exception.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition in file **LanguageException.java**.

8.7 workspace/S-TGDH_god/manager/ILauncher.java File Reference

Launcher interface. Each launcher must implement this interface.

Namespaces

- namespace **manager**

Classes

- interface **manager::ILauncher**
Launcher(p. 67) *interface. Each launcher must implement this interface.*

8.7.1 Detailed Description

Launcher interface. Each launcher must implement this interface.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition in file **ILauncher.java**.

8.8 workspace/S-TGDH_god/manager/Log.java File Reference

Logging class.

Namespaces

- namespace **manager**

Classes

- class **manager::Log**
Logging class.

8.8.1 Detailed Description

Logging class.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/24

Definition in file **Log.java**.

8.9 workspace/S-TGDH_god/manager/messages/Message.java File Reference

Unicast message stored in the network boxes.

Namespaces

- namespace `manager::messages`

Classes

- class `manager::messages::Message`
Unicast message stored in the network boxes.

8.9.1 Detailed Description

Unicast message stored in the network boxes.

See also:

`manager.Supervisor::broadcastGroup(INode, String, String, String)`

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file `Message.java`.

8.10 workspace/S-TGDH_god/manager/messages/Multicast-Message.java File Reference

Multicast message stored in the network boxes.

Namespaces

- namespace **manager::messages**

Classes

- class **manager::messages::MulticastMessage**
Multicast message stored in the network boxes.

8.10.1 Detailed Description

Multicast message stored in the network boxes.

See also:

`manager.Supervisor::broadcastGroup(INode, String, String, String)`

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file **MulticastMessage.java**.

8.11 workspace/S-TGDH_god/manager/Supervisor.java File Reference

Network modelization.

Namespaces

- namespace **manager**
- namespace **java::util**

Classes

- class **manager::Supervisor**
Network modelization.

8.11.1 Detailed Description

Network modelization.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition in file **Supervisor.java**.

8.12 workspace/S-TGDH_god/manager/Supervisor-Message.java File Reference

Messages modelisation for the interaction between the nodes (and the network).

Namespaces

- namespace **manager**

Classes

- interface **manager::SupervisorMessage**
Messages modelisation for the interaction between the nodes (and the network).

8.12.1 Detailed Description

Messages modelisation for the interaction between the nodes (and the network).

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition in file **SupervisorMessage.java**.

8.13 workspace/S-TGDH_god/namespace.doxydoc File Reference

Main page for the documentation.

Namespaces

- namespace **console**
- namespace **lang**
- namespace **manager**
- namespace **manager::messages**
- namespace **node**
- namespace **node::buffers**
- namespace **node::messages**
- namespace **tools**
- namespace **ui**

8.13.1 Detailed Description

Main page for the documentation.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file **namespace.doxydoc**.

8.14 workspace/S-TGDH_god/node/buffers/Challenge2Response.java File Reference

Memorize the challenge (challenge 2) to send to which node when all the nodes will have send a ANSWER_GROUP_MESSAGE.

Namespaces

- namespace **node::buffers**

Classes

- class **node::buffers::Challenge2Response**

Memorize the challenge (challenge 2) to send to which node when all the nodes will have send a ANSWER_GROUP_MESSAGE.

8.14.1 Detailed Description

Memorize the challenge (challenge 2) to send to which node when all the nodes will have send a ANSWER_GROUP_MESSAGE.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file **Challenge2Response.java**.

8.15 workspace/S-TGDH_god/node/buffers/ChallengeExpected.java File Reference

Memorize the challenge (challenge 1) associated to all the members of the group.

Namespaces

- namespace `node::buffers`

Classes

- class `node::buffers::ChallengeExpected`
Memorize the challenge (challenge 1) associated to all the members of the group.

8.15.1 Detailed Description

Memorize the challenge (challenge 1) associated to all the members of the group.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file `ChallengeExpected.java`.

8.16 workspace/S-TGDH_god/node/DelayException.java File Reference

Exception to manage unordered messages.

Namespaces

- namespace **node**

Classes

- class **node::DelayException**
Exception to manage unordered messages.

8.16.1 Detailed Description

Exception to manage unordered messages.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file **DelayException.java**.

8.17 workspace/S-TGDH_god/node/GroupNode.java File Reference

Model the group behavior of a node.

Namespaces

- namespace **node**

Classes

- class **node::GroupNode**
Model the group behavior of a node.

8.17.1 Detailed Description

Model the group behavior of a node.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file **GroupNode.java**.

8.18 workspace/S-TGDH_god/node/IGroupNode.java File Reference

Interface for the group behavior of a node.

Namespaces

- namespace **node**

Classes

- interface **node::IGroupNode**
Interface for the group behavior of a node.

8.18.1 Detailed Description

Interface for the group behavior of a node.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file **IGroupNode.java**.

8.19 workspace/S-TGDH_god/node/IInitNode.java File Reference

Interface for the group initiation behavior of a node.

Namespaces

- namespace **node**

Classes

- interface **node::IInitNode**
Interface for the group initiation behavior of a node.

8.19.1 Detailed Description

Interface for the group initiation behavior of a node.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file **IInitNode.java**.

8.20 workspace/S-TGDH_god/node/InitNode.java File Reference

Model the group initiation behavior of a node.

Namespaces

- namespace **node**

Classes

- class **node::InitNode**
Model the group initiation behavior of a node.

8.20.1 Detailed Description

Model the group initiation behavior of a node.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file **InitNode.java**.

8.21 workspace/S-TGDH_god/node/INode.java File Reference

Model the default behavior of a node.

Namespaces

- namespace **node**

Classes

- interface **node::INode**
Model the default behavior of a node.

8.21.1 Detailed Description

Model the default behavior of a node.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file **INode.java**.

8.22 workspace/S-TGDH_god/node/messages/Node-Message.java File Reference

Modelize a message in the node stack.

Namespaces

- namespace `node::messages`

Classes

- class `node::messages::NodeMessage`
Modelize a message in the node stack.

8.22.1 Detailed Description

Modelize a message in the node stack.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file `NodeMessage.java`.

8.23 workspace/S-TGDH_god/node/Node.java File Reference

Modelisation of a node of the network.

Namespaces

- namespace **node**

Classes

- class **node::Node**
Modelisation of a node of the network.

8.23.1 Detailed Description

Modelisation of a node of the network.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/22

Definition in file **Node.java**.

8.24 workspace/S-TGDH_god/node/NodeBehavior.java File Reference

Model the standard behavior of a node.

Namespaces

- namespace **node**

Classes

- class **node::NodeBehavior**
Model the standard behavior of a node.

8.24.1 Detailed Description

Model the standard behavior of a node.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file **NodeBehavior.java**.

8.25 workspace/S-TGDH_god/node/NodeConstants.java File Reference

Interface for the nodes: contains standard constants.

Namespaces

- namespace **node**

Classes

- interface **node::NodeConstants**
Interface for the nodes: contains standard constants.

8.25.1 Detailed Description

Interface for the nodes: contains standard constants.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/26

Definition in file **NodeConstants.java**.

8.26 workspace/S-TGDH_god/tools/Crypto.java File Reference

Centralized the commonly used functions to perform cryptographic operations.

Namespaces

- namespace **tools**

Classes

- class **tools::Crypto**
Centralized the commonly used functions to perform cryptographic operations.

8.26.1 Detailed Description

Centralized the commonly used functions to perform cryptographic operations.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition in file **Crypto.java**.

8.27 workspace/S-TGDH_god/tools/JavaTools.java File Reference

Centralized the commonly used functions to perform Java operations.

Namespaces

- namespace **tools**

Classes

- class **tools::JavaTools**

Centralized the commonly used functions to perform Java operations.

8.27.1 Detailed Description

Centralized the commonly used functions to perform Java operations.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition in file **JavaTools.java**.

8.28 workspace/S-TGDH_god/tools/Tree.java File Reference

Represents the cryptographic tree of the (S-)TGDH algorithm.

Namespaces

- namespace **tools**

Classes

- class **tools::Tree**
Represents the cryptographic tree of the (S-)TGDH algorithm.

8.28.1 Detailed Description

Represents the cryptographic tree of the (S-)TGDH algorithm.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition in file **Tree.java**.

8.29 workspace/S-TGDH_god/tools/TreeNode.java File Reference

Represents the standard element of the Tree structure.

Namespaces

- namespace **tools**

Classes

- class **tools::TreeNode**

*Represents the standard element of the **Tree**(p.102) structure.*

8.29.1 Detailed Description

Represents the standard element of the Tree structure.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition in file **TreeNode.java**.

8.30 workspace/S-TGDH_god/ui/About.java File Reference

About window of the project.

Namespaces

- namespace **ui**

Classes

- class **ui::About**
About(p. 21) *window of the project.*

8.30.1 Detailed Description

About window of the project.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition in file **About.java**.

8.31 workspace/S-TGDH_god/ui/CellStyle.java File Reference

Define a cell style used by XGridBag.

Namespaces

- namespace **ui**

Classes

- class **ui::CellStyle**
Define a cell style used by XGridBag(p. 113).

8.31.1 Detailed Description

Define a cell style used by XGridBag.

Date:

2007/06/13

See also:

XGridBag
java.awt.GridBagConstraints

Version:

1.0

Author:

//Web source//

Definition in file **CellStyle.java**.

8.32 workspace/S-TGDH_god/ui/Constants.java File Reference

Graphical constants for the project components.

Namespaces

- namespace **ui**

Classes

- interface **ui::Constants**
Graphical constants for the project components.

8.32.1 Detailed Description

Graphical constants for the project components.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition in file **Constants.java**.

8.33 workspace/S-TGDH_god/ui/CreateGroup.java File Reference

Configuration window for a group creation.

Namespaces

- namespace **ui**

Classes

- class **ui::CreateGroup**
Configuration window for a group creation.

8.33.1 Detailed Description

Configuration window for a group creation.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition in file **CreateGroup.java**.

8.34 workspace/S-TGDH_god/ui/Interface.java File Reference

Communication interface for graphical mode of the simulator.

Namespaces

- namespace **ui**

Classes

- class **ui::Interface**
Communication interface for graphical mode of the simulator.

8.34.1 Detailed Description

Communication interface for graphical mode of the simulator.

Author:

Julien Thomas

Version:

1.0

Date:

2007/07/09

Definition in file **Interface.java**.

8.35 workspace/S-TGDH_god/ui/Manager.java File Reference

Core window of the simulator graphical mode.

Namespaces

- namespace **ui**

Classes

- class **ui::Manager**
Core window of the simulator graphical mode.

8.35.1 Detailed Description

Core window of the simulator graphical mode.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition in file **Manager.java**.

8.36 workspace/S-TGDH_god/ui/NodeFrame.java File Reference

Window use to retrieve information on a node of the network.

Namespaces

- namespace **ui**

Classes

- class **ui::NodeFrame**

Window use to retrieve information on a node of the network.

8.36.1 Detailed Description

Window use to retrieve information on a node of the network.

Author:

Julien Thomas

Version:

1.0

Date:

2007/06/23

Definition in file **NodeFrame.java**.

8.37 workspace/S-TGDH_god/ui/XGridBag.java File Reference

The XGridBag helps to use the GridBagConstraints.

This class defines styles reusable by differents components in a grid.

Namespaces

- namespace **ui**

Classes

- class **ui::XGridBag**

*The **XGridBag**(p.113) helps to use the GridBagConstraints.*

This class defines styles reusable by differents components in a grid.

8.37.1 Detailed Description

The XGridBag helps to use the GridBagConstraints.

This class defines styles reusable by differents components in a grid.

To create a layout do the following step:

- create a GridBagLayout and a XGridBag.
- prepare the components size (min/prefered/max).
- create one or more style using `CellStyle.getStyle` method
- add components to the grid using the `XGridBag.add` method.

Example:

```
GridBagLayout layout = new GridBagLayout();
panel.setLayout(layout);
XGridBag grid = new XGridBag(panel);
CellStyle style1 = new CellStyle(1.0, 0.0,
GridBagConstraints.WEST, GridBagConstraints.VERTICAL, insets, 4, 0);
...
grid.add(comp1, style1, 0, 0);
grid.add(comp2, style1, 0, 1);
grid.add(comp3, style2, 1, 0);
..
```

See also:

CellStyle
java.awt.GridBagConstraints
java.awt.Insets

Date:

2007/06/13

Version:

1.0

Author:

//Web source//

Definition in file **XGridBag.java**.