

Class `Frame_Client`

```
java.lang.Object
|
+--java.awt.Component
    |
    +--java.awt.Container
        |
        +--java.awt.Window
            |
            +--java.awt.Frame
                |
                +--javax.swing.JFrame
                    |
                    +--Frame_Client
```

All Implemented Interfaces:

`javax.accessibility.Accessible`, `java.awt.image.ImageObserver`, `java.awt.MenuContainer`, `javax.swing.RootPaneContainer`, `java.io.Serializable`, `javax.swing.WindowConstants`

public class **Frame_Client**
extends `javax.swing.JFrame`

Title: SWE - 1. Abgabe Description: RMI-Client

Version:

0.1

This document is copyright (C) 2001 Gerald Haider, and it's free. You can distribute it under the terms of the GNU General Public License, which you can get at <http://www.gnu.org/copyleft/gpl.html>

Author:

Gerald Haider

<http://www.krikkit.net>

See Also:

[Serialized Form](#)

Method Summary

private void	<code>jbInit()</code>
(package private) void	<code> jButton clear actionPerformed</code> (<code>java.awt.event.ActionEvent e</code>)
(package private) void	<code> jButton getAll actionPerformed</code> (<code>java.awt.event.ActionEvent e</code>) methode invokes the RMI-methode <code>getAll</code> to get all messages on the RMI server ruft die RMI - Methode <code>getAll</code> auf und gibt alle Nachrichten im logging fenster aus
(package private) void	<code> jButton getMsg actionPerformed</code> (<code>java.awt.event.ActionEvent e</code>) methode invokes the RMI-methode <code>getMsg</code> to get messages from a RMI server aufruf der RMI-Methode <code>getMsg</code> , falls Nachrichten vorhanden sind werden diese dem Benutzer angezeigt, falls nicht erhaelt er eine Fehlermeldung mit dem Hinweis das keine Nachrichten für ihn vorhanden sind.
(package private) void	<code> jButton sendMsg actionPerformed</code> (<code>java.awt.event.ActionEvent e</code>) methode invokes the RMI-methode <code>sendMsg</code> to send messages to a RMI server ruft die RMI-Methode <code>sendMsg</code> auf, vor deren aufruf wird aber überprüft ob die jeweiligen textfelder genug text enthalten und erst dann wird der String-Array mit den Empfängern aufgebaut und danach die RMI-Methoden <code>sendMsg</code> aufgerufen.

protected void	processWindowEvent (java.awt.event.WindowEvent e)
-------------------	---

Method Detail

jbInit

```
private void jbInit()  
    throws java.lang.Exception
```

processWindowEvent

```
protected void processWindowEvent(java.awt.event.WindowEvent e)
```

Overrides:

processWindowEvent in class javax.swing.JFrame

jButton_getMsg_actionPerformed

```
void jButton_getMsg_actionPerformed(java.awt.event.ActionEvent e)
```

methode invokes the RMI-methode getMsg to get messages from a RMI server aufruf der RMI-Methode getMsg, falls Nachrichten vorhanden sind werden diese dem Benutzer angezeigt, falls nicht erhaelt er eine Fehlermeldung mit dem Hinweis das keine Nachrichten für ihn vorhanden sind.

Parameters:

e - ActionEvent

jButton_sendMsg_actionPerformed

```
void jButton_sendMsg_actionPerformed(java.awt.event.ActionEvent e)
```

methode invokes the RMI-methode sendMsg to send messages to a RMI server ruft die RMI-Methode sendMsg auf, vor deren aufruf wird aber überprüft ob die jeweiligen textfelder genug text enthalten und erst dann wird der String-Array mit den Empfängern aufgebaut und danach die RMI-Methoden sendMsg aufgerufen.

Parameters:

e - ActionEvent

jButton_getAll_actionPerformed

```
void jButton_getAll_actionPerformed(java.awt.event.ActionEvent e)
```

methode invokes the RMI-methode getAll to get all messages on the RMI server ruft die RMI - Methode getAll auf und gibt alle Nachrichten im logging fenster aus

Parameters:

e - ActionEvent

Class **RMIClient**

java.lang.Object

|
+--**RMIClient**

public class **RMIClient**
extends java.lang.Object

Title: SWE - 1. Abgabe Description: RMI-Client

Version:

0.1

This document is copyright (C) 2001 Gerald Haider, and it's free. You can distribute it under the terms of the GNU General Public License, which you can get at

<http://www.gnu.org/copyleft/gpl.html>

Author:

Gerald Haider

<http://www.krikkit.net>

Field Detail

packFrame

boolean **packFrame**

Constructor Detail

RMIClient

public **RMIClient**()

Method Detail

main

public static void **main**(java.lang.String[] args)

Class SimpleRMIServer

```
java.lang.Object
|
+--java.rmi.server.RemoteObject
    |
    +--java.rmi.server.RemoteServer
        |
        +--java.rmi.server.UnicastRemoteObject
            |
            +--SimpleRMIServer
```

All Implemented Interfaces:

java.rmi.Remote, java.io.Serializable, [SimpleRMIInterface](#)

```
public class SimpleRMIServer
extends java.rmi.server.UnicastRemoteObject
implements SimpleRMIInterface
```

Title: SWE - 1. Abgabe Description: RMI-Server

Version:

0.1

This document is copyright (C) 2001 Gerald Haider, and it's free. You can distribute it under the terms of the GNU General Public License, which you can get at

<http://www.gnu.org/copyleft/gpl.html>

Author:

Gerald Haider

<http://www.krikkit.net>

See Also:

[Serialized Form](#)

Method Summary

java.lang.String	getAll () gives back all messages in the server queue diese Methode wurde eigentlich nur zu Testzwecken implementiert, sie macht im Prinzip das selbe wie getMsg nur werden hier ALLE Nachrichten ALLER User ausgegeben
java.lang.String[]	getMsg (java.lang.String user) gets a message for the given user from server queue beim abrufen der nachrichten wird der methoder getMsg nur als String der Name der Benutzers übergeben.
static void	main (java.lang.String[] argv)
void	sendMsg (java.lang.String user, java.lang.String[] to, java.lang.String msg) sends a message to the message server queue der methode werden ein String mit den Usernamen der Senders, ein String-Array mit den Namen der Empfänger und ein String mit dem zu sendenen Text übergeben.

Constructor Detail

SimpleRMIServer

```
public SimpleRMIServer()  
    throws java.rmi.RemoteException
```

Method Detail

sendMsg

```
public void sendMsg(java.lang.String user,  
                   java.lang.String[] to,  
                   java.lang.String msg)
```

sends a message to the message server queue der methode werden ein String mit den Usernamen der Senders, ein String-Array mit den Namen der Empfänger und ein String mit dem zu sendenen Text übergeben. Die Speicherung der Daten erfolgt in einem Hashtable, wobei der username als key dient und die nachrichten werden in einem Vector gespeichert der in ebendiesem Hashtable ist falls eine Nachricht an einen noch nicht vorhandenen user gerichtet ist wird einfach ein eintrag fuer ihn im hashtable angelegt, falls er bereits vorhanden ist wird die neue Nachricht einfach an den Vector angehängt

Specified by:

[sendMsg](#) in interface [SimpleRMIInterface](#)

Parameters:

`user` - the name the sender
`to` - an String-array with all the recipients
`msg` - the message to send

getMsg

```
public java.lang.String[] getMsg(java.lang.String user)
```

gets a message for the given user from server queue beim abrufen der nachrichten wird der Methode getMsg nur als String der Name der Benutzers übergeben. Die Nachrichten werden mittels eines Schleifendurchlaufs aus der Speicherstruktur (Hashtable - Vector) extrahiert und in einen zusammenhängenden String zusammengefasst und zurueckgegeben.

Specified by:

[getMsg](#) in interface [SimpleRMIInterface](#)

Parameters:

`user` - the name the user

Returns:

an String-Array with all the messages for the given user

getAll

```
public java.lang.String getAll()
```

gives back all messages in the server queue diese Methode wurde eigentlich nur zu Testzwecken implementiert, sie macht im Prinzip das selbe wie getMsg nur werden hier ALLE Nachrichten ALLER User ausgegeben

Specified by:

[getAll](#) in interface [SimpleRMIInterface](#)

Returns:

a String with all messages on the server