Table Of Content

Class Des

```
java.lang.Object
    |
    +--SecureChat.crypto.Des
```

< Constructors > < Methods >

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

Constructors

Des

```
public  Des(javax.crypto.SecretKey key)
```

Main Constructor

Parameters:

key - : A secret shared key

Methods

DesDecrypt

```
public java.lang.String DesDecrypt(byte[] EncryptData)
                        throws java.security.NoSuchAlgorithmException,
                                  null,
                                  null,
                                  null,
                                  javax.crypto.BadPaddingException
```

Allows to decrypt a message according des algorithm

Parameters:

EncryptData - : Data to decrypt

Returns:

String : Data decrypted

Throws:

java.security.NoSuchAlgorithmException -
null -
null -
null -
javax.crypto.BadPaddingException -

DesEncrypt

```
public byte[] DesEncrypt(java.lang.String data)
              throws java.security.NoSuchAlgorithmException,
                          null,
                          null,
                          null,
                          javax.crypto.BadPaddingException
```

Allows to encrypt a message according des algorithm

Parameters:

data - : Data to encrypt

Returns:

byte[] : Data encypted

Throws:

java.security.NoSuchAlgorithmException -
null -
null -
null -
javax.crypto.BadPaddingException -

getSessionKey

```
public javax.crypto.SecretKey getSessionKey()
```

Get the shared key

Returns:

SecretKey

Class DiffieHellman

```
java.lang.Object
    |
    +--SecureChat.crypto.DiffieHellman
```

< Constructors > < Methods >

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

Constructors

DiffieHellman

```
public  DiffieHellman(Rsa rsa)
```

Main constructor

Parameters:

rsa - : An rsa object used to sign data

Methods

genKeystream

```
public javax.crypto.SecretKey genKeystream(java.lang.String PathBase,
                                           java.io.ObjectOutputStream
StreamOut,
                                           java.io.ObjectInputStream StreamIn,
                                           java.lang.String FName)
                                    throws java.io.IOException,
                                           java.security.SignatureException
```

This function allows to generate a secret key using Diffie-Hellman algorithm

Parameters:

PathBase - : Path of file wich contains P and Q numbers to use in Diffie-Hellman
StreamOut - : A socket stream (out)
StreamIn - : A socket stream (in)
FName - : The friend name whereby main user is talking

Returns:

SecretKey : A shared secret key to use for the talk session

Throws:

java.io.IOException -
java.security.SignatureException -

---

isValid

```
public boolean isValid()
```

Allows to check whether an istance of this class is valid or not

Returns:

boolean: True if the istance is valid, false otherwise

---

readBaseKey

```
public void readBaseKey(java.lang.String PathBase)
            throws java.io.IOException
```

This function reads the two numbers P and Q used by Diffie-Hellman

Parameters:

PathBase - : Path where the file is stored

Throws:

java.io.IOException -

Class Rsa

```
java.lang.Object
    |
    +--SecureChat.crypto.Rsa
```

< Constructors > < Methods >

---

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

Constructors

Rsa

```
public  Rsa(java.lang.String KeyDirectory,
            SecureLogin login)
```

Main constructor

Parameters:

login - : Used to identify the owner of rsa keys
KeyDirectory - : Directory of the keys

Methods

CheckSign

```
public boolean CheckSign(byte[] message,
                         byte[] sign,
                         java.lang.String UserName)
```

GetPublicKey

```
public java.security.PublicKey GetPublicKey(java.lang.String UserName)
                                throws java.io.IOException,
java.security.spec.InvalidKeySpecException,
                                                    null
```

Gets a public key stored giving the username

Parameters:

UserName - : User name

Returns:

PublicKey

Throws:

java.io.IOException -
java.security.spec.InvalidKeySpecException -
null -

SignMessage

```
public byte[] SignMessage(byte[] message)
```

---

createKeys

```
public void createKeys()
            throws java.io.IOException,
                        java.security.NoSuchAlgorithmException
```

This function allows to create a pair of RSA keys which will be stored in two different file

Throws:

java.io.IOException -
java.security.NoSuchAlgorithmException -

---

isPresent

```
public boolean isPresent(java.lang.String UserName)
```

Checks whether a public key is present

Parameters:

UserName - : User name

Returns:

boolean : True if present, false otherwise

---

setKeyDirectory

`public void setKeyDirectory(java.lang.String KeyDirectory)`

Sets the keys directory

Parameters:

KeyDirectory - : Path so set as default

---

setUserName

`public boolean setUserName(java.lang.String name)`

Set the keys owner this function has called after a login in order to protect the key from others on the same computer

Parameters:

name - : Name of the user

Returns:

boolean : True whether the user has already logged

Class Rsa.KEY

```
java.lang.Object
    |
    +--java.lang.Enum
        |
        +--SecureChat.crypto.Rsa.KEY
```

All Implemented Interfaces:

java.io.Serializable, java.lang.Comparable

< Fields > < Methods >

---

public static final class **Rsa.KEY**
extends java.lang.Enum

Enum

Fields

PRIVATE

`public static final Rsa.KEY PRIVATE`

PUBLIC

```
public static final Rsa.KEY PUBLIC
```

Methods

valueOf

```
public static Rsa.KEY valueOf(java.lang.String name)
```

---

values

```
public static SecureChat.crypto.Rsa.KEY[] values()
```

Class Directory

```
java.lang.Object
    |
    +--SecureChat.file.Directory
```

< Fields > < Constructors > < Methods >

---

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

Fields

LOCALHOST

```
public static final java.lang.String LOCALHOST
```

Constructors

Directory

```
public  Directory()
```

Methods

MakeDirectory

```
public static void MakeDirectory(java.lang.String path)
                  throws java.io.IOException
```

Allows to create a new directory

Parameters:

path - : path of directory

Throws:

java.io.IOException -

Class Path

```
java.lang.Object
    |
    +--SecureChat.file.Path
```

< Fields > < Constructors >

---

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

Fields

CREDENTIALSPATH

```
public java.lang.String CREDENTIALSPATH
```

Users credentials path

KEYDIRECTORY

```
public java.lang.String KEYDIRECTORY
```

Key directory path

PATHDH

```
public java.lang.String PATHDH
```

Diffie-Hellman base numbers path

Constructors

Path

```
public  Path()
```

Main constructor

Class Menu

```
java.lang.Object
    |
    +--SecureChat.graphics.Menu
```

< Constructors > < Methods >

---

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

Constructors

Menu

```
public  Menu()
```

Methods

ChatBoard

```
public void ChatBoard(java.lang.String toShow)
```

Pritns the main chat board screen

Parameters:

toShow - : additional informations to show

InitialMenu

```
public static int InitialMenu()
                  throws java.io.IOException
```

Prints a simple initial menu whereby user can choose what to do

Returns:

int : User choice (1 login , 0 registration)

Throws:

java.io.IOException -

NewUserMenu

```
public java.lang.String[] NewUserMenu()
                          throws java.io.IOException
```

Prints a simple registration menu

Returns:

String[] : Credentials (in order user name and password)

Throws:

java.io.IOException -

RegisteredMenu

```
public java.lang.String[] RegisteredMenu()
                          throws java.io.IOException
```

Prints a Login menu

Returns:

String[] : Credentials (in order user name and password)

Throws:

java.io.IOException -

Class SecureLogin

```
java.lang.Object
    |
    +--SecureChat.login.SecureLogin
```

< Constructors > < Methods >

---

public class **SecureLogin**

extends java.lang.Object

Constructors

SecureLogin

```
public  SecureLogin()
```

Main constructor

Methods

LoadUser

```
public boolean LoadUser(java.lang.String UserName,
                        java.lang.String password,
                        Path path)
                 throws java.io.IOException,
                        java.io.FileNotFoundException
```

Checks if user credentials are valids

Parameters:

UserName - : user name
password - : password
path - : path of credentials

Returns:

boolean : true whether the credentials given are correct, false otherwise

Throws:

java.io.IOException -
java.io.FileNotFoundException -

newUser

```
public boolean newUser(java.lang.String UserName,
                       java.lang.String password,
                       Path path)
                throws java.io.IOException,
                       java.io.FileNotFoundException,
                       java.security.NoSuchAlgorithmException,
                       java.io.UnsupportedEncodingException
```

Puts a new user in the users list

Parameters:

UserName - : user name
password - : password
path - : path of credentials

Returns:

boolean : true whether everything has gone well, false otherwise

Throws:

java.io.IOException -
java.io.FileNotFoundException -
java.security.NoSuchAlgorithmException -
java.io.UnsupportedEncodingException -

---

userBound

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

Gets name of user bound to the class instance

Returns:

String: the user name

---

userLogged

```
public boolean userLogged()
```

Check whether the user has logged successfully

Returns:

boolean: result of checks

Class User

```
java.lang.Object
    |
    +--SecureChat.login.User
```

< Constructors > < Methods >

---

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

Constructors

User

```
public  User(int port,
             java.lang.String server,
             SecureLogin log)
```

Main constructor

Parameters:

port - : port whereby the user is connecting
server - : Server Ip
log - : A secure login istance

Methods

CreateRsa

```
public boolean CreateRsa(java.lang.String KeyDir)
               throws java.io.IOException,
                         java.security.NoSuchAlgorithmException
```

Allows to create a pair Rsa keys

Parameters:

KeyDir - : directory where the keys will be stored

Returns:

boolean : True whether everything has gone well

Throws:

java.io.IOException -
java.security.NoSuchAlgorithmException -

Decrypt

```
public java.lang.String Decrypt(byte[] data)
                         throws java.security.NoSuchAlgorithmException,
                                null,
                                null,
                                null,
                                javax.crypto.BadPaddingException
```

Allows to decrypt a message according des algorithm

Parameters:

data - : Data to decrypt

Returns:

String : Data decrypted

Throws:

java.security.NoSuchAlgorithmException -
null -
null -
null -
javax.crypto.BadPaddingException -

Encrypt

```
public byte[] Encrypt(java.lang.String data)
              throws java.security.NoSuchAlgorithmException,
                     null,
                     null,
                     null,
                     javax.crypto.BadPaddingException
```

Allows to encrypt a message according des algorithm

Parameters:

data - : Data to encrypt

Returns:

byte[] : Data encypted

Throws:

java.security.NoSuchAlgorithmException -
null -
null -
null -
javax.crypto.BadPaddingException -

---

createDiffieHellman

```
public javax.crypto.SecretKey createDiffieHellman(java.lang.String path,
                                                   java.io.ObjectOutputStream
StreamOut,
                                                   java.io.ObjectInputStream
ois)
                                throws java.io.IOException,
java.security.SignatureException
```

Allows to use Diffie-Hellman's protocol

Parameters:

path - : Path of the main variables to use
StreamOut - : A socket data stream (out)
ois - : A socket data stream (in)

Returns:

SecretKey : Secret key

Throws:

java.io.IOException -
java.security.SignatureException -

---

desInstance

```
public boolean desInstance(javax.crypto.SecretKey key)
```

Allows to create a des istance

Parameters:

key - : Secret shared key

Returns:

boolean : True if everything has gone well

---

getClientIp

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

Gets the client port

Returns:

port: port

---

getClientPort

`public int` **`getClientPort`**`()`

Gets the client port

Returns:

port: port

---

getFriendName

`public java.lang.String` **`getFriendName`**`()`

Retrieves name user whereby the main user is talking

Returns:

String: User name

---

getServerIp

`public java.lang.String` **`getServerIp`**`()`

---

getServerPort

`public int` **`getServerPort`**`()`

Gets the server port

Returns:

port: port

---

getUserName

`public java.lang.String` **`getUserName`**`()`

Retrieves name of main user

Returns:

String: User name

---

isRsaPresent

`public boolean isRsaPresent(java.lang.String UserName)`

Checks whether Rsa public key is present or not

Parameters:

UserName - : Key owner

Returns:

boolean : True if the key is present

---

isValid

`public boolean isValid()`

Checks wheter the class istance is valid or not

Returns:

boolean: validity

---

setClientIp

`public void setClientIp(java.lang.String ip)`

Sets the client ip

Parameters:

ip - : ip

---

setClientPort

`public void setClientPort(int port)`

Sets the client port

Parameters:

port - : port

---

setFriendName

`public void setFriendName(java.lang.String name)`

Sets the name of user whereby the main user is talking

Parameters:

name - : Friend name

setServerIp

`public void` **`setServerIp`**`(java.lang.String server)`

sets the server ip

Parameters:

server - : ip

---

setServerPort

`public void` **`setServerPort`**`(int port)`

sets the server port

Parameters:

port - : port

INDEX