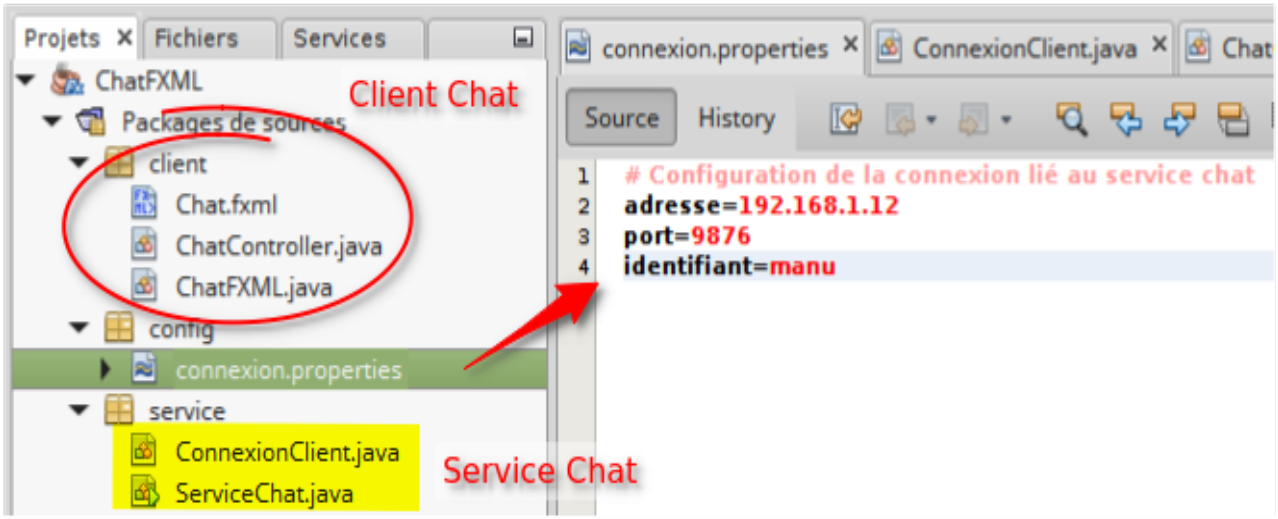
/home/manu/CloudStation/ProjetsJAVA/ChatFXML/src/service/ServiceChat.java

```
package service;

import java.io.*;
import java.net.*;

public class ServiceChat {
  private ServiceChat() {
    try {
      ServerSocket service = new ServerSocket(9876);
      System.out.println("Service en fonctionnement");
      while (true) {
        Socket client = service.accept();
        new ConnexionClient(client).start();
      }
    }
    catch (IOException ex) {  System.err.println("Le service existe déjà"); }
  }

  public static void main(String[] args) { new ServiceChat(); }
}
```



/home/manu/CloudStation/ProjetsJAVA/ChatFXML/src/service/ConnexionClient.java

```
package service;

import java.io.*;
import java.net.*;
import java.util.*;
import java.util.Map.Entry;

class ConnexionClient extends Thread {
  private Scanner requête;
  private String login, destinataire, message;
  private static HashMap<String, PrintWriter> clients = new HashMap<>();

  public ConnexionClient(Socket client) throws IOException {
    requête = new Scanner(client.getInputStream());
    PrintWriter réponse = new PrintWriter(client.getOutputStream(), true);
    login = requête.next();
    clients.put(login, réponse);
    System.out.println(login+" connecté");
    listeDesConnectés(true);
  }

  @Override
  public void run() {
    while (true) {
      destinataire = requête.next();
      System.out.println(destinataire);
      message = requête.nextLine();
      System.out.println(message);
      if (message.equals(" stop")) break;
      clients.get(destinataire).println(login+" > "+message);
    }
    System.out.println(login+" déconnecté");
    clients.get(destinataire).println("stop");
    clients.remove(login);
    listeDesConnectés(false);
  }

  private void listeDesConnectés(boolean connecté) {
    for (Entry<String, PrintWriter> entrée : clients.entrySet()) {
      entrée.getValue().println(login + " vient de se "+(connecté ? "connecter":"déconnecter
      entrée.getValue().println("Connectés : "+clients.keySet().toString());
    }
  }
}
```

/home/manu/CloudStation/ProjetsJAVA/ChatFXML/src/client/ChatFXML.java

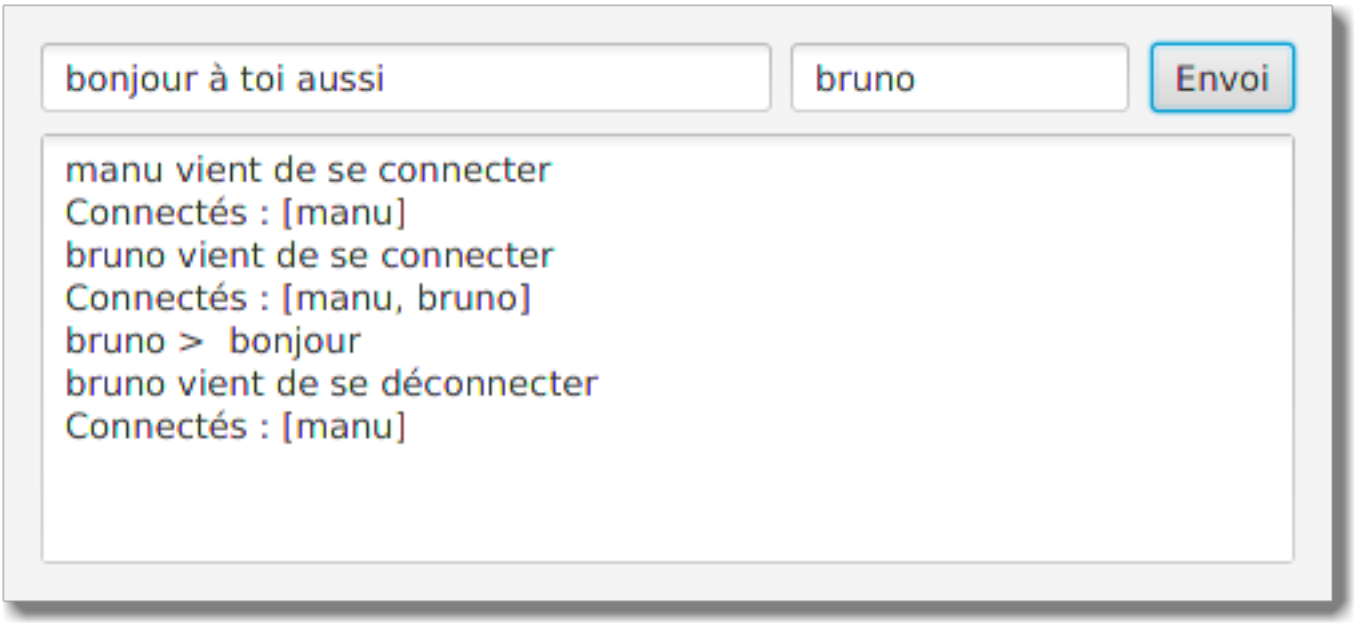
```
package client;

import javafx.application.Application;
import javafx.fxml.*;
import javafx.scene.*;
import javafx.stage.Stage;

public class ChatFXML extends Application {
  private ChatController contrôleur;

  @Override
  public void start(Stage fenêtre) throws Exception {
    FXMLLoader chargeurFXML = new FXMLLoader();
    chargeurFXML.setLocation(getClass().getResource("Chat.fxml"));
    Parent root = chargeurFXML.load();
    contrôleur = chargeurFXML.getController();
    Scene scene = new Scene(root);
    fenêtre.setScene(scene);
    fenêtre.setTitle("Chat");
    fenêtre.show();
  }

  @Override
  public void stop() throws Exception {
    contrôleur.arrêt();
    super.stop();
  }
}
```

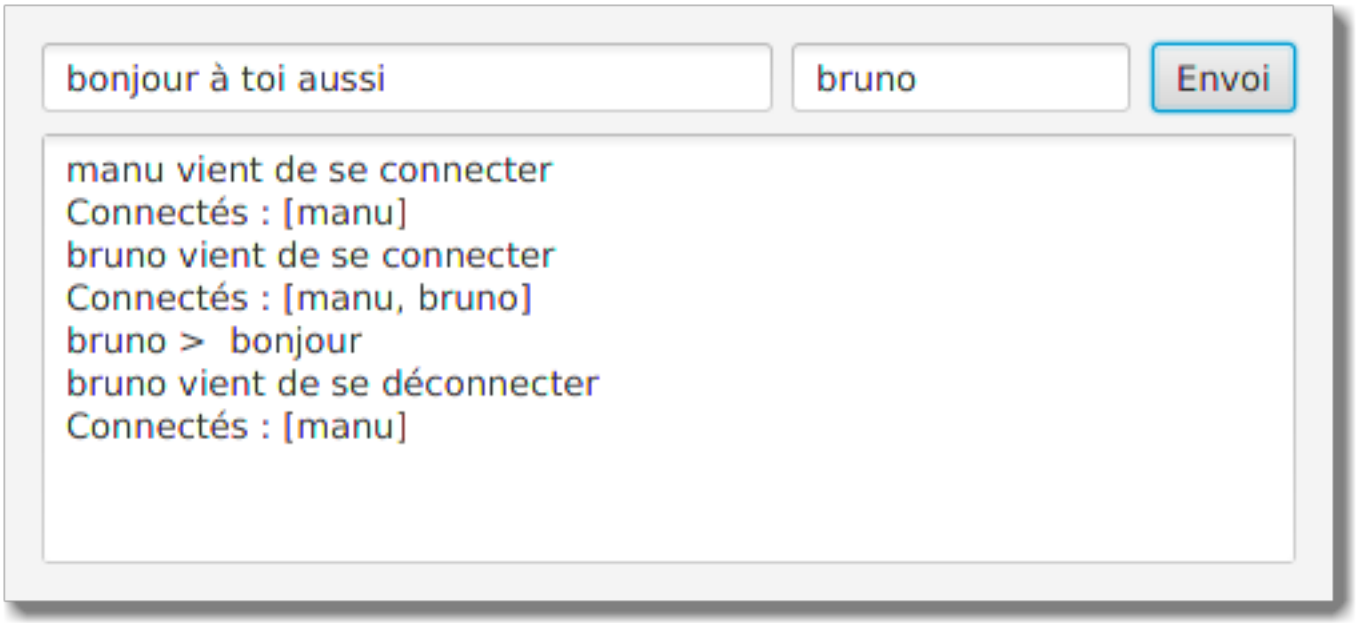


/home/manu/CloudStation/ProjetsJAVA/ChatFXML/src/client/Chat.fxml

```
<?xml version="1.0" encoding="UTF-8"?>

<?import java.lang.*?>
<?import java.util.*?>
<?import javafx.scene.*?>
<?import javafx.scene.control.*?>
<?import javafx.scene.layout.*?>

<AnchorPane prefHeight="380.0" prefWidth="536.0"
        xmlns:fx="http://javafx.com/fxml/1" xmlns="http://javafx.com/javafx/8" fx:controller="client.ChatController">
  <children>
    <Button layoutX="468.0" layoutY="14.0" onAction="#soumettre" text="Envoi"
        AnchorPane.rightAnchor="14.0" AnchorPane.topAnchor="14.0" />
    <TextField fx:id="message" layoutX="14.0" layoutY="14.0" prefHeight="25.0" prefWidth="312.0"
        promptText="Message" AnchorPane.leftAnchor="14.0"
        AnchorPane.rightAnchor="210.0" AnchorPane.topAnchor="14.0" />
    <TextField fx:id="destinataire" layoutX="333.0" layoutY="14.0" prefHeight="25.0" prefWidth="127.0"
        promptText="Destinataire" AnchorPane.rightAnchor="76.0" AnchorPane.topAnchor="14.0" />
    <TextArea fx:id="zone" layoutX="14.0" layoutY="48.0" prefHeight="322.0" prefWidth="512.0"
        AnchorPane.bottomAnchor="14.0" AnchorPane.leftAnchor="14.0"
        AnchorPane.rightAnchor="14.0" AnchorPane.topAnchor="48.0" />
  </children>
</AnchorPane>
```



/home/manu/CloudStation/ProjetsJAVA/ChatFXML/src/client/ChatController.java

```
package client;

import java.io.*;
import java.net.*;
import java.util.*;
import javafx.fxml.*;
import javafx.scene.control.*;

public class ChatController implements Initializable {
  @FXML private TextField message;
  @FXML private TextField destinataire;
  @FXML private TextArea zone;

  private Scanner réponse;
  private PrintWriter envoi;
  private String identifiant;

  @FXML
  private void soumettre() {
    envoi.println(destinataire.getText()+' '+message.getText());
  }

  public void arrêt() {
    envoi.println(identifiant+" stop");
  }

  @Override
  public void initialize(URL url, ResourceBundle rb) {
    ResourceBundle config = ResourceBundle.getBundle("config/connexion");
    String adresse = config.getString("adresse");
    int port = Integer.parseInt(config.getString("port"));
    identifiant = config.getString("identifiant");
    try {
      Socket service = new Socket(adresse, port);
      envoi = new PrintWriter(service.getOutputStream(), true);
      envoi.println(identifiant);
      réponse = new Scanner(service.getInputStream());
      zone.appendText(réponse.nextLine()+"\n");
      new Thread() {
        @Override
        public void run() {
          while (true) {
            String message = réponse.nextLine();
            if (message.equals("stop")) break;
            zone.appendText(message+"\n");
          }
        }
      }.start();
    }
    catch (IOException ex) {
      Alert problème = new Alert(Alert.AlertType.WARNING);
      problème.setHeaderText("Impossible de se connecter");
      problème.showAndWait();
    }
  }
}
```

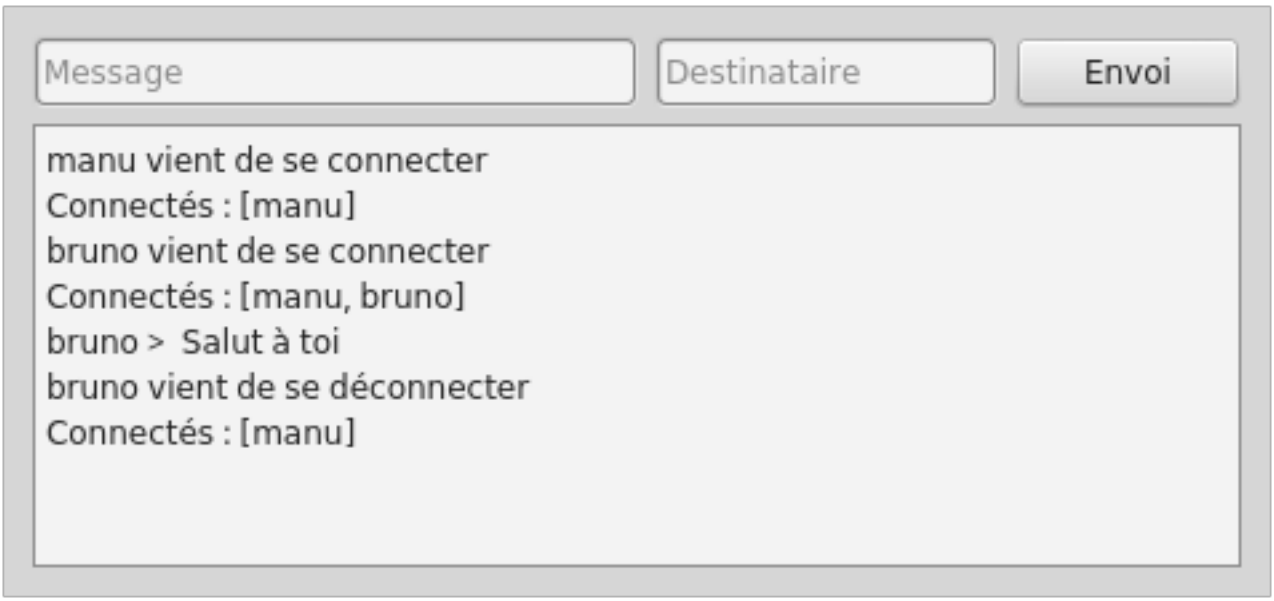
```
#ifndef PRINCIPAL_H
#define PRINCIPAL_H

#include "ui_principal.h"
#include <QtNetwork>

class Principal : public QMainWindow, private Ui::Principal
{
  Q_OBJECT

public:
  explicit Principal(QWidget *parent = 0);
private slots:
  void soumettre();
  void connexion();
  void reception();
private:
  void configuration();
protected:
  void closeEvent(QCloseEvent *cloture);
private:
  QTcpSocket service;
  QString adresse, identifiant;
  QTextStream requete;
  int port;
};

#endif // PRINCIPAL_H
```



```cpp
#include "principal.h"
#include <QCloseEvent>

Principal::Principal(QWidget *parent) : QMainWindow(parent), requete(&service)
{
  setupUi(this);
  connect(&service, SIGNAL(readyRead()), this, SLOT(reception()));
  connect(&service, SIGNAL(connected()), this, SLOT(connexion()));
  configuration();
  service.connectToHost(adresse, port);
}

void Principal::configuration()
{
  QSettings config("connexion.ini", QSettings::IniFormat);
  adresse = config.value("localisation/adresse").toString();
  port = config.value("localisation/port").toInt();
  identifiant = config.value("localisation/identifiant").toString();
}

void Principal::connexion()
{
  requete << identifiant << endl;
}

void Principal::soumettre()
{
  requete << destinataire->text() << ' ' << message->text() << endl;
}

void Principal::closeEvent(QCloseEvent *cloture)
{
  requete << identifiant << " stop" << endl;
  cloture->accept();
}

void Principal::reception()
{
  while (service.canReadLine())
  {
    QByteArray texte = service.readLine();
    texte.remove(texte.size()-1, 1);
    zone->append(texte);
  }
}
```