מכללת אורט ק. ביאליק

פ ר ו י י ק ט ג מ ר

למילוי חלקי של הדרישות לקבלת תואר

ה נ ד ס א י

ה נ ד ס ת - ת ו כ נ ה

בהתמחות : מ ח ש ב י ם

נושא הפרוייקט : מעבדת תקשורת

שם הסטודנט/ית : אלון ישראל אנגל

העבודה בוצעה בהנחיית : מוטי פניקשווילי ואודי מלכה

שנה"ל תשע"ט 2019

כתובת : רח' ניר עם 3, פינת הגליל (ת"ד 340) , קרית ביאליק 27103

טלפון : 04-8490091 , פקס : 04-8424856 , מייל : kb.ort.org.il

מכללת אורט קרית ביאליק לטכנולוגיה מתקדמת ולמדעים

הצהרת סטודנט

אני הסטודנט אלון ישראל אנגל מס' ת.ז 208015032 החתום מטה, מצהיר בזאת שכל עבודת הגמר/ הפרוייקט המוגש/ת בחוברת זו היינו/ה פרי עבודתי בלבד.

על בסיס הנחייתו של המנחה ותוך הסתמכות על מקורות הידע והמידע האחרים המצויים בביליוגרפיה המובאת בחוברת זאת.

אני מודע לאחריות שהנני מקבל על עצמי ע"י חתימתי על הצהרה זו שכל הנאמר בה הינו אמת ורק אמת.

חתימת מגיש החוברת : _____

<u>אישור המנחה :</u>

הנני מאשר הגשת החוברת להערכה _____

Contents

הקדמה

בפרויקט המעבדה מוצג קשר בין צד שרת לצד לקוח, בפרויקט שלי אעסוק עם RSS Feed.

RSS הוא קיצור לראשי תיבות Rich Site Summary או Really Simple Syndication, בעצם RSS הוא תמצות את האתר לתמלול מסכם של הפרטים בתוך האתר.

לתמצות האתר יש שימושים רבים בין אם הם לנגישות לכבדי ראייה לאתרים או שילוב באתרים אחרים (סרגל חדשות לדוגמא שמופיע באתר). או בשביל הודעות פוש בטלפון לאפליקציות מסוימות.

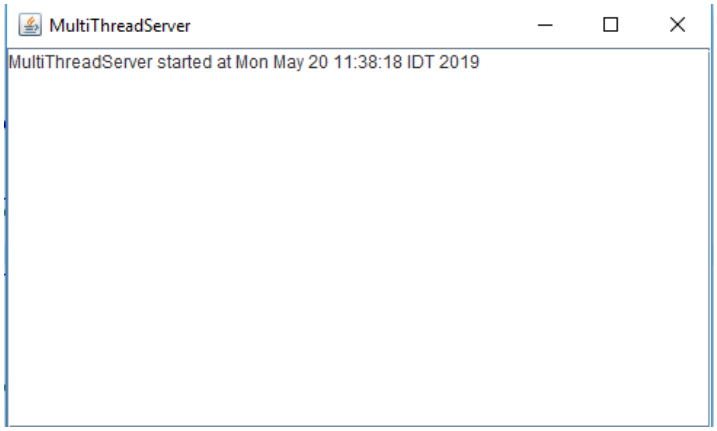
בפרויקט אדגים קשר בין צד השרת לצד הלקוח באמצעות הזנת קישור לעמוד RSS מצד הלקוח אל צד השרת וקבלת תשובה מצד השרת בהתאם לתנאים מסוימים בהם אם נעשה שימוש בקישור או שלא נעשה. (ניתן לעשות שימוש בקישור רק פעם אחת בלבד – ככה שלא תהיה עוד פעם נוספת).

בפרויקט נעשה שימוש בפעולת שהיינה synchronized על מנת שכאשר לקוח אחד משתמש בסרבר לקוח שני ימתין כדי שלא יווצר מצב ששניים יגשו לאותה רשימה ויערכו אותה באותו זמן (כך שלא יווצר מצב שבמקרה קיצון שני לקוחות עם אותו קישור יוכלו לעשות שימוש בו, אלא רק הראשון שניגש מביניהם).

הפעלת התוכנה:

1. ראשית הפעל את הסרבר

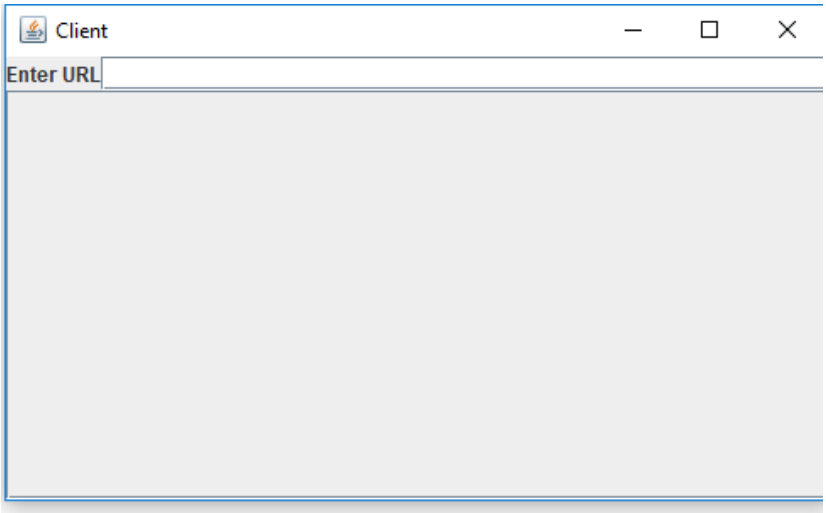
-הרץ את המחלקה MultiThreadServerGUI ובעקבות זאת יופיע החלון הבא:



(למעלה- חלון הסרבר).

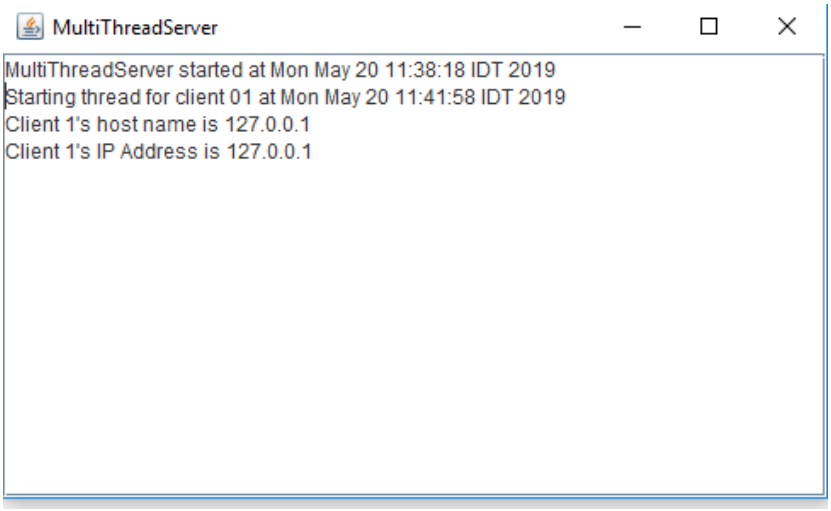
2. הרץ את הצד לקוח

- הרץ את המחלקה ClientGui ובעקבות זאת יופיע החלון הבא:



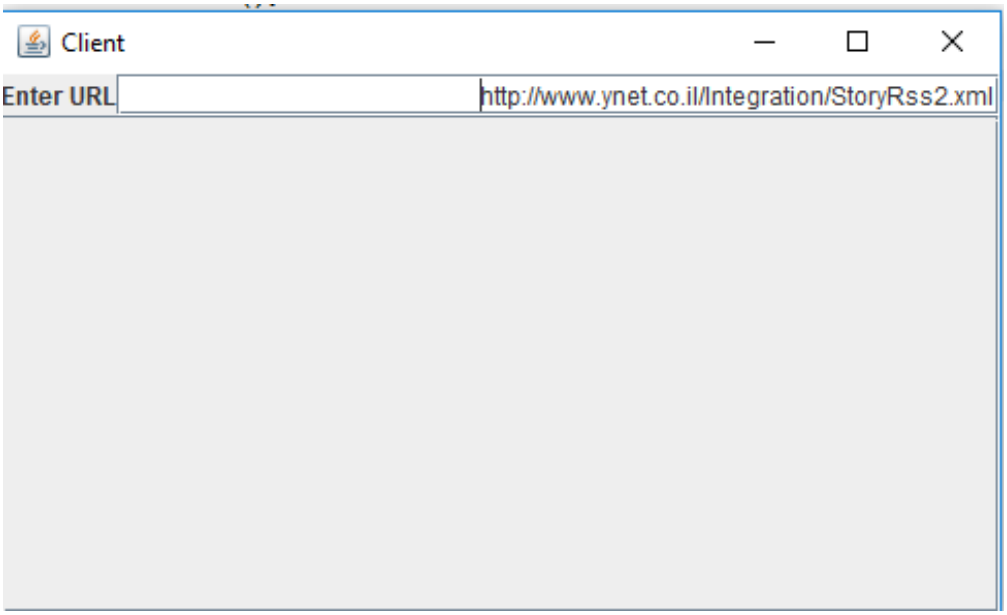
(למעלה – חלון הקליינט).

כאשר לקוח מתחבר על הסרבר הוא מופיע בו:

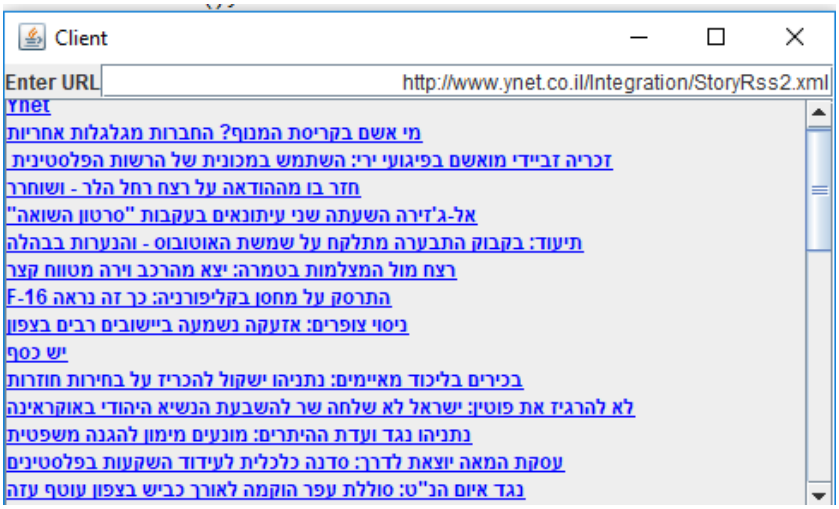


(במקרה זה הקליינט הוא הראשון).

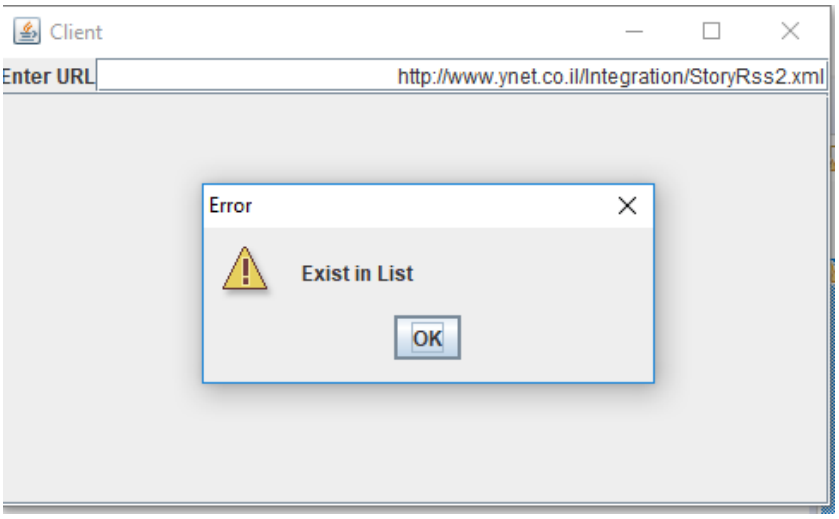
על מנת להכניס קשר יש לכתוב \ להדביקו בתיבת הטקסט ולאחר מכן ללחוץ על Enter במקלדת בחלון הקליינט.



לאחר לחיצה על כפתור ה-Enter במידה והקישור הוא קישור RSS לאתר יוצגו כל ה-RSS Feed של האתר ועל ידי לחיצה על הטקסט ניתן לגשת לאותם כתבות:

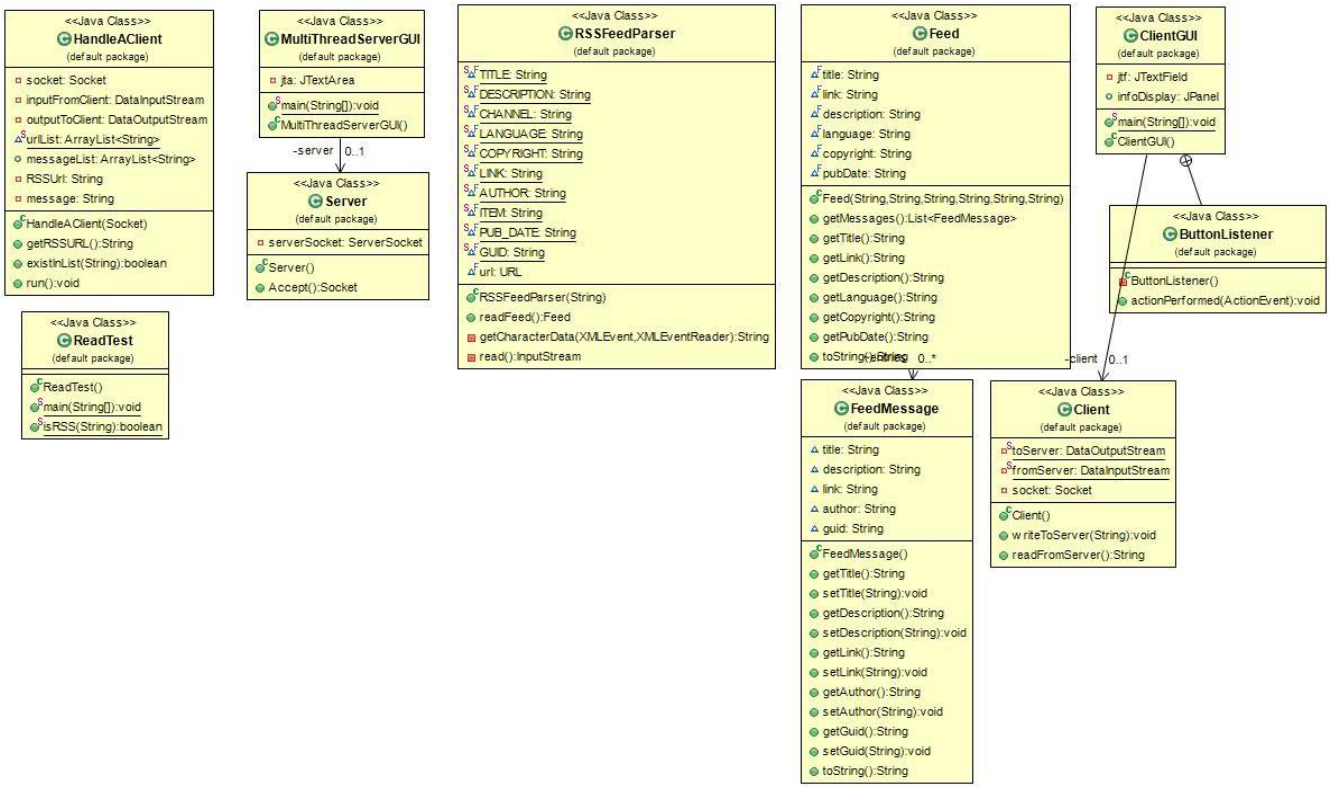


במידה ובקישור נעשה שימוש על ידי אחד המשתמשים בסרבר יופיע החלון הבא:



במידה ונוסיף קישור נוסף המשך ה-RSS Feed יופיע מתחת למה שכבר הוסף.

תרשים UML:


<<Java Class>>
HandleAClient
(default package)
socket: Socket
inputFromClient: DataInputStream
outputToClient: DataOutputStream
urlList: ArrayList<String>
messageList: ArrayList<String>
RSSUrl: String
message: String
HandleAClient(Socket)
getRSSURL():String
existInList(String):boolean
run():void
<<Java Class>>
ReadTest
(default package)
ReadTest()
main(String[]):void
isRSS(String):boolean
<<Java Class>>
MultiThreadServerGUI
(default package)
jta: JTextArea
main(String[]):void
MultiThreadServerGUI()
-server
0..1
<<Java Class>>
Server
(default package)
serverSocket: ServerSocket
Server()
Accept():Socket
<<Java Class>>
RSSFeedParser
(default package)
TITLE: String
DESCRIPTION: String
CHANNEL: String
LANGUAGE: String
COPYRIGHT: String
LINK: String
AUTHOR: String
ITEM: String
PUB_DATE: String
GUID: String
url: URL
RSSFeedParser(String)
readFeed():Feed
getCharacterData(XMLEvent,XMLEventReader):String
read():InputStream
<<Java Class>>
Feed
(default package)
title: String
link: String
description: String
language: String
copyright: String
pubDate: String
Feed(String,String,String,String,String,String)
getMessages():List<FeedMessage>
getTitle():String
getLink():String
getDescription():String
getLanguage():String
getCopyright():String
getPubDate():String
toString():String
-feeds
0..*
-client
0..1
<<Java Class>>
ClientGUI
(default package)
jtf: JTextField
infoDisplay: JPanel
main(String[]):void
ClientGUI()
<<Java Class>>
ButtonListener
(default package)
ButtonListener()
actionPerformed(ActionEvent):void
<<Java Class>>
FeedMessage
(default package)
title: String
description: String
link: String
author: String
guid: String
FeedMessage()
getTitle():String
setTitle(String):void
getDescription():String
setDescription(String):void
getLink():String
setLink(String):void
getAuthor():String
setAuthor(String):void
getGuid():String
setGuid(String):void
toString():String
<<Java Class>>
Client
(default package)
toServer: DataOutputStream
fromServer: DataInputStream
socket: Socket
Client()
writeToServer(String):void
readFromServer():String

פירוט מחלקות:

מחלקת ClientGui:

מחלקה גרפית של המשתמש עבור התוכנה.

פירוט	שדה
שדה טקסט לכתיבת הקישור (URL)	`private JTextField jtf`
משתנה מסוג קלינט (משתמש).	`private Client client`
פאנל הצגת המידע מן ה- RSS Feed של הקישור.	`public JPanel infoDisplay`

פירוט	שיטה
שיטה הנקראת כאשר המשתמש מקליד	`private void processMessage(String message)`
פעולה שמתבצעת כאשר נלחץ המקש הנבחר (במקרה שלנו זה תקשורת עם השרת וקבלת מידע מהשרת).	`public void actionPerformed`

מחלקת Server:

שרת עבור התוכנה.

פירוט	שדה
סוקט לחיבורי משתמשים חדשים	`private ServerSocket serversocket`

פירוט	שיטה
ממתין לחיבור שיבוצע לסוקט ומקבל אותו.	`public Socket Accept()`
פעולה בונה.	`public Server()`

מחלקת MultiThreadServerGUI:

מחלקה גרפית של השרת היוצרת תהליך לשליחת הודעות ממשתמשים.

פירוט	שדה
שדה טקסט	`private JTextArea jta`
משתנה מסוג סרבר (בניית עצם).	`private Server server = new Server();`

פירוט	שיטה
פעולה בונה היוצרת תהליכים לפי כמות הסוקטים למשתמשים.	`public MultiThreadServerGUI()`

מחלקת HandleAClient:

מחלקת עזר המטפלת בפנייה מהמשתמש אל השרת.

פירוט	שדה
סוקט לחיבורים.	`private Socket socket`
זרם קלט הנתונים	`private DataInputStream inputFromClient`
זרם פלט הנתונים.	`private DataOutputStream outputToClient;`
רשימת מחרוזות הקישורים שנעשו בהם שימוש.	`static ArrayList<String> urlList = new ArrayList<>();`
רשימת ההודעות (לשליחה למשתמשים את כל ה-Feed).	`public ArrayList<String> messageList = new ArrayList<>();`
כתובת הקישור	`private String RSSUrl;`
מחרוזת ההודעה שתשלח למשתמש.	`private String message;`

פירוט	שיטה
פעולה בונה.	`public HandleAClient(Socket socket)`
פעולה המחזירה את כתובת הקישור (URL)	`public String getRSSURL()`
פעולה המחזירה האם המחרוזת קיימת ברשימה או לא.	`public synchronized boolean existInList(String url)`
הפעולה מבצעת תהליך לשליחת הודעות שהתקבלו מהסוקט (משתמש) אל השרת ושולחת את תגובת השרת לקישור אל המשתמש	`public void run()`

מחלקת RSSFeedParser:

מחלקה אשר אחראית על ניתוח ה-RSS Feed מהקישור.

פירוט	שדה
משתנה מסוג מחרוזת של כותרת	`static final String TITLE`
משתנה מסוג מחרוזת של תיאור	`static final String DESCRIPTION`
משתנה מסוג מחרוזת של ערוץ	`static final String CHANNEL`
משתנה מסוג מחרוזת של שפה	`static final String LANGUAGE`
משתנה מסוג מחרוזת של זכויות יוצרים	`static final String COPYRIGHT`
משתנה מסוג מחרוזת של קישור	`static final String LINK`
משתנה מסוג מחרוזת של כותב	`static final String AUTHOR`
משתנה מסוג מחרוזת של פריט	`static final String ITEM`
משתנה מסוג מחרוזת של תאריך שחרור	`static final String PUB_DATE`
משתנה מסוג מחרוזת של מזהה יחודי	`static final String GUID`
כתובת ה-URL	`final URL url;`

פירוט	שיטה
פעולה המגדירה את הקישור	`public RSSFeedParser(String feedUrl)`
פעולה הקוראת את ה-Feed	`public Feed readFeed()`
פעולה הקוראת את התווים	`private String getCharacterData(XMLEvent event, XMLEventReader eventReader)`
פעולה הקוראת את סטרים הקלט	`private InputStream read()`

מחלקת FeedMessage:

מחלקה האחראית על הגדרות של משתני הודעת ה-Feed.

פירוט:	שדה:
משתנה מסוג מחרוזת של כותרת	`String title`
משתנה מסוג מחרוזת של תיאור	`String description`
משתנה מסוג מחרוזת של קישור (הלינק).	`String link`
משתנה מסוג מחרוזת של כותב	`String author`
משתנה מסוג מחרוזת של מזהה יחודי	`String guid`

פירוט	שיטה
פעולה המחזירה את הכותרת	`public String getTitle()`

פעולה המגדירה את הכותרת	`public void setTitle(String title)`
פעולה המחזירה את תיאור	`public String getDescription()`
פעולה המגדירה את תיאור	`public void setDescription(String description)`
פעולה המחזירה את הלינק	`public String getLink()`
פעולה המגדירה את הלינק	`public void setLink(String link)`
פעולה המחזירה את המחבר	`public String getAuthor()`
פעולה המגדירה את המחבר	`public void setAuthor(String author)`
פעולה המחזירה את המזהה היחודי	`public String getGuid()`
פעולה המגדירה את המזהה היחודי	`public void setGuid(String guid)`
toString	`public String toString()`

מחלקת Feed:

מחלקה האחראית על הפעולה הבונה של ה-Feed.

פירוט:	**שדה:**
משתנה מסוג מחרוזת של כותרת	`String title`
משתנה מסוג מחרוזת של קישור (הלינק).	`String link`
משתנה מסוג מחרוזת של תיאור	`String description`
משתנה מסוג מחרוזת של תיאור השפה	`String language`
משתנה מסוג מחרוזת של זכויות היוצרים	`String copyright`
משתנה מסוג מחרוזת של תאריך השחרור.	`String pubDate`
משתנה מסוג רשימה של הכניסות.	`final List<FeedMessage> entries`

פירוט	**שיטה**
פעולה בונה	`public Feed(String title, String link, String description, String language, String copyright, String pubDate)`
פעולה המחזירה את ההודעות	`public List<FeedMessage> getMessages()`
פעולה המחזירה את הכותרת	`public String getTitle()`
פעולה המחזירה את תיאור	`public String getDescription()`
פעולה המחזירה את תיאור השפה.	`public String getLanguage()`

פעולה המחזירה את זכויות היוצרים של ה-Feed	`public String getCopyright()`
פעולה המחזירה את תאריך השחרור \ הפצה.	`public String getPubDate()`
toString	`public String toString()`

מחלקת Client:

מחלקת המשתמש של התוכנה.

פירוט	שדה
הסוקט שמחבר בין הלקוח לשרת	`private Socket socket`
תקשורת עם הסרבר כלפי חוץ	`private static DataOutputStream toServer`
תקשורת עם הסרבר כלפי פנים	`private static DataInputStream fromServer`

פירוט	שיטה
פעולה בונה המגדירה את הסוקט ומקבלת ושולחת מידע אל הסרבר ומהסרבר.	`public Client()`
פעולה הכותבת ושולחת אל הסרבר את הקישור.	`public void writeToServer(String st)`
פעולה הקוראת את ההודעות מהסרבר	`public String readFromServer()`

קטעי קוד:

JAVA:

מחלקת ClientGUI:

```
1. import java.awt.BorderLayout;
2. import java.awt.Desktop;
3. import java.awt.GridLayout;
4. import java.awt.event.ActionEvent;
5. import java.awt.event.ActionListener;
6. import java.awt.event.MouseAdapter;
7. import java.awt.event.MouseEvent;
8. import java.io.IOException;
9. import java.net.URI;
10.  import java.net.URISyntaxException;
11.
12.  import javax.swing.JFrame;
13.  import javax.swing.JLabel;
14.  import javax.swing.JOptionPane;
15.  import javax.swing.JPanel;
16.  import javax.swing.JScrollPane;
17.  import javax.swing.JTextField;
18.
19.  public class ClientGUI extends JFrame {
20.          private JTextField jtf = new JTextField();
21.          private Client client = new Client();
22.          public JPanel infoDisplay = new JPanel(new GridLayout(0,
   1));
23.
24.          public static void main(String[] args) {
25.                  new ClientGUI();
26.          }
27.
28.          public ClientGUI() {
29.                  // The Graphics of the client (URL TextArea nd the
   display of the info).
30.                  JPanel p = new JPanel();
31.                  p.setLayout(new BorderLayout());
32.                  p.add(new JLabel("Enter URL"), BorderLayout.WEST);
33.                  p.add(jtf, BorderLayout.CENTER);
34.
35.                  jtf.setHorizontalAlignment(JTextField.RIGHT);
36.                  setLayout(new BorderLayout());
37.                  add(p, BorderLayout.NORTH);
38.                  add(new JScrollPane(infoDisplay),
   BorderLayout.CENTER);
39.                  jtf.addActionListener(new ButtonListener());
40.                  setTitle("Client");
41.                  setSize(500, 300);
42.                  setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
43.                  setVisible(true);
44.          }
45.
46.          private class ButtonListener implements ActionListener {
47.                  public void actionPerformed(ActionEvent e) {
48.                          String urlRSS = jtf.getText();
49.
```

```
50.                         // Send the url to the server
51.                         client.writeToServer(urlRSS);
52.
53.                         // Get message from the server
54.                         String message = client.readFromServer();
55.
56.
57.                         System.out.println(message);
58.                         if (message.contains("Exist in List")) {
59.                                 //Open Message Window Because the url
   exist in the lists already (been used).
60.                                 JOptionPane.showMessageDialog(null,
   "Exist in List", "Error", JOptionPane.WARNING_MESSAGE);
61.
62.                         }
63.
64.                         //The url isn't used yet.
65.                         if (message != "Exist in List") {
66.                                 String[] splitStrToArticles =
   message.split("FeedMessage");
67.
68.
69.                                 for (int i = 1; i <
   splitStrToArticles.length; i++) {
70.                                         String url =
   splitStrToArticles[i].split("link=")[1].split(", author=")[0];
71.                                         String name =
   splitStrToArticles[i].split("title=")[1].split(",")[0];
72.                                         String str = "<html><a
   href=\"" + url + "\" >" + name + "</a><html>" + "\n";
73.                                         JLabel info = new
   JLabel(str);
74.                                         System.out.println(str);
75.                                         info.addMouseListener(new
   MouseAdapter() {
76.
77.                                                 @Override
78.                                                 //In Case mouse
   clicked on one of the displayed lables, open the article in the
   browser
79.                                                 public void
   mouseClicked(MouseEvent e) {
80.                                                         try {
81.
     Desktop.getDesktop().browse(new URI(url));
82.                                                         } catch
   (IOException | URISyntaxException e1) {
83.
     e1.printStackTrace();
84.                                                         }
85.                                                 }
86.
87.                                         });

88.                                         infoDisplay.add(info);
89.                                 }
90.                         }
91.                         revalidate();
92.
93.                 }
94.         }
```

```
95.  }
```

מחלקת Server:

```
1. import java.io.IOException;
2. import java.net.ServerSocket;
3. import java.net.Socket;
4.
5. class Server {
6.
7.    private ServerSocket serverSocket;
8.
9.    public Server() {
10.        try {
11.           serverSocket = new ServerSocket(8000);
12.      } catch (IOException e) {
13.         e.printStackTrace();
14.         }
15.     }
16.
17.     public Socket Accept() {
18.          try {
19.           return serverSocket.accept();
20.           } catch (IOException e) {
21.                  e.printStackTrace();
22.           }
23.        return null;
24.     }
25.
26.  }
```

מחלקת MultiThreadServerGUI:

```
1. import java.io.*;
2. import java.net.*;
3. import java.util.*;
4. import java.awt.*;
5. import javax.swing.*;
6.
7. public class MultiThreadServerGUI extends JFrame {
8.
9.   private JTextArea jta = new JTextArea();
10.         private Server server = new Server();
11.
12.         public static void main(String[] args) {
13.                 new MultiThreadServerGUI();
14.         }
15.
16.         public MultiThreadServerGUI() {
17.                 // The text area on the frame
18.                 setLayout(new BorderLayout());
19.                 add(new JScrollPane(jta), BorderLayout.CENTER);
20.                 setTitle("MultiThreadServer");
```

```
21.                 setSize(500, 300);
22.                 setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
23.                 setVisible(true); // Must show the frame in the GUI
24.
25.
26.                 jta.append("MultiThreadServer started at " + new
   Date() + '\n');
27.
28.                 int clientNo = 1;
29.
30.                 while (true) {
31.                         // Listen for a new connection request.
32.                         Socket socket = server.Accept();
33.
34.                         // Display the client number
35.                         jta.append("Starting thread for client 0" +
   clientNo + " at " + new Date() + '\n');
36.
37.                         // Find the client's host name, and IP
   address
38.                         InetAddress inetAddress =
   socket.getInetAddress();
39.                         jta.append("Client " + clientNo + "'s host
   name is " + inetAddress.getHostName() + "\n");
40.                         jta.append("Client " + clientNo + "'s IP
   Address is " + inetAddress.getHostAddress() + "\n");
41.
42.                         // Create a new task for the connection
43.                         Thread task = new Thread(new
   HandleAClient(socket));
44.                         task.start();
45.                         clientNo++;
46.                 }
47.         }
48.  }
```

מחלקת HandleAClient:

```
1. import java.io.DataInputStream;
2. import java.io.DataOutputStream;
3. import java.io.IOException;
4. import java.io.ObjectOutputStream;
5. import java.net.Socket;
6. import java.util.ArrayList;
7. import java.util.List;
8.
9. class HandleAClient implements Runnable {
10.         private Socket socket; // A connected socket
11.
12.         private DataInputStream inputFromClient;
13.         private DataOutputStream outputToClient;
14.
15.         static ArrayList<String> urlList = new ArrayList<>();
16.         public ArrayList<String> messageList = new ArrayList<>();
17.         private String RSSUrl;
18.         private String message;
19.
20.
```

```
21.         //Constructor
22.         public HandleAClient(Socket socket) {
23.                 this.socket = socket;
24.         }
25.         //Return the RSSURL
26.         public String getRSSURL() {
27.                 return RSSUrl;
28.         }
29.         //Return if string exist in the list or not (false \ true)
30.         public synchronized boolean existInList(String url) {
31.
32.                 if (urlList.contains(url)) {
33.
34.                         return true;
35.                 }
36.
37.                 return false;
38.         }
39.
40.
41.         public void run() {
42.                 try {
43.                         // Create data input and output streams
44.                         inputFromClient = new
   DataInputStream(socket.getInputStream());
45.                         outputToClient = new
   DataOutputStream(socket.getOutputStream());
46.
47.                         // work non stop until false (when no
   client).
48.                         while (true) {
49.                                 // Receive RSSURL from the client
50.
51.                                 RSSUrl = inputFromClient.readUTF();
52.                                 if (urlList != null) {
53.                                         //URL LIST is Not Empty
54.                                         if (existInList(RSSUrl)) {
55.                                                 //URL that Already
   Been USED
56.                                                 message = "Exist in
   List";
57.
     System.out.println("Exist");
58.
     outputToClient.writeUTF(message);
59.
60.                                         } else {
61.                                                 //NEW URL to the
   existing list.
62.
     System.out.println(urlList.toString());
63.
     System.out.println("Dont Exist 1");
64.                                                 urlList.add(RSSUrl);
65.
66.                                                 RSSFeedParser parser
   = new RSSFeedParser(RSSUrl);
67.                                                 Feed feed =
   parser.readFeed();
68.                                                 StringBuilder
   stringBuilder = new StringBuilder();
```

```
69.
70.
71.
     System.out.println(feed);
72.                                                 for (FeedMessage
   messageF : feed.getMessages()) {
73.
     stringBuilder.append(String.valueOf(messageF) + "\n");
74.
     System.out.println(messageF);
75.
76.                                                 }
77.
78.                                                 message =
   stringBuilder.toString();
79.
     outputToClient.writeUTF(message);
80.
81.                                         }
82.
83.                                 } else {
84.                                         //List IS Empty (in the start
   of the server) - so adding the URL to the list.
85.                                         System.out.println("Dont
   exist 2");
86.                                         urlList.add(RSSUrl);
87.                                         message = RSSUrl;
88.
     outputToClient.writeUTF(message);
89.
90.                                 }
91.
92.
93.
94.                         }
95.                 } catch (IOException e) {
96.                         System.err.println(e);
97.                 }
98.         }
99.  }
```

מחלקת RSSFeedParser:

```
1. import java.io.IOException;
2. import java.io.InputStream;
3. import java.net.MalformedURLException;
4. import java.net.URL;
5.
6. import javax.xml.stream.XMLEventReader;
7. import javax.xml.stream.XMLInputFactory;
8. import javax.xml.stream.XMLStreamException;
9. import javax.xml.stream.events.Characters;
10.  import javax.xml.stream.events.XMLEvent;
11.
12.
13.  public class RSSFeedParser {
```

```
14.      static final String TITLE = "title";
15.      static final String DESCRIPTION = "description";
16.      static final String CHANNEL = "channel";
17.      static final String LANGUAGE = "language";
18.      static final String COPYRIGHT = "copyright";
19.      static final String LINK = "link";
20.      static final String AUTHOR = "author";
21.      static final String ITEM = "item";
22.      static final String PUB_DATE = "pubDate";
23.      static final String GUID = "guid";
24.
25.      final URL url;
26.
27.      //Set the url as the current url.
28.      public RSSFeedParser(String feedUrl) {
29.          try {
30.              this.url = new URL(feedUrl);
31.          } catch (MalformedURLException e) {
32.              throw new RuntimeException(e);
33.          }
34.      }
35.      //Reads The Feed
36.      public Feed readFeed() {
37.          Feed feed = null;
38.          try {
39.              boolean isFeedHeader = true;
40.              // Set header values intial to the empty string
41.              String description = "";
42.              String title = "";
43.              String link = "";
44.              String language = "";
45.              String copyright = "";
46.              String author = "";
47.              String pubdate = "";
48.              String guid = "";
49.
50.              // First create a new XMLInputFactory
51.              XMLInputFactory inputFactory =
   XMLInputFactory.newInstance();
52.              // Setup a new eventReader
53.              InputStream in = read();
54.              XMLEventReader eventReader =
   inputFactory.createXMLEventReader(in);
55.              // read the XML document
56.              while (eventReader.hasNext()) {
57.                  XMLEvent event = eventReader.nextEvent();
58.                  if (event.isStartElement()) {
59.                      String localPart =
   event.asStartElement().getName()
60.                              .getLocalPart();
61.                      switch (localPart) {
62.                      case ITEM:
63.                          if (isFeedHeader) {
64.                              isFeedHeader = false;
65.                              feed = new Feed(title, link,
   description, language,
66.                                      copyright, pubdate);
67.                          }
68.                          event = eventReader.nextEvent();
69.                          break;
70.                      case TITLE:
```

```
71.                         title = getCharacterData(event,
   eventReader);
72.                         break;
73.                     case DESCRIPTION:
74.                         description = getCharacterData(event,
   eventReader);
75.                         break;
76.                     case LINK:
77.                         link = getCharacterData(event,
   eventReader);
78.                         break;
79.                     case GUID:
80.                         guid = getCharacterData(event,
   eventReader);
81.                         break;
82.                     case LANGUAGE:
83.                         language = getCharacterData(event,
   eventReader);
84.                         break;
85.                     case AUTHOR:
86.                         author = getCharacterData(event,
   eventReader);
87.                         break;
88.                     case PUB_DATE:
89.                         pubdate = getCharacterData(event,
   eventReader);
90.                         break;
91.                     case COPYRIGHT:
92.                         copyright = getCharacterData(event,
   eventReader);
93.                         break;
94.                     }
95.                 } else if (event.isEndElement()) {
96.                     if
   (event.asEndElement().getName().getLocalPart() == (ITEM)) {
97.                         FeedMessage message = new FeedMessage();
98.                         message.setAuthor(author);
99.                         message.setDescription(description);
100.                        message.setGuid(guid);
101.                        message.setLink(link);
102.                        message.setTitle(title);
103.                        feed.getMessages().add(message);
104.                        event = eventReader.nextEvent();
105.                        continue;
106.                    }
107.                }
108.            }
109.        } catch (XMLStreamException e) {
110.            throw new RuntimeException(e);
111.        }
112.        return feed;
113.    }
114.
115.    private String getCharacterData(XMLEvent event, XMLEventReader
   eventReader)
116.            throws XMLStreamException {
117.        String result = "";
118.        event = eventReader.nextEvent();
119.        if (event instanceof Characters) {
120.            result = event.asCharacters().getData();
121.        }
```

```
        return result;
    }
    //Read the feed (input stream).
    private InputStream read() {
        try {
            return url.openStream();
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
}
```

מחלקת ReadTest:

```
import java.io.IOException;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import javax.xml.parsers.ParserConfigurationException;

import org.w3c.dom.Document;
import org.xml.sax.SAXException;
//Class For Display of the RSS feed
 public class RssReadTest {
        // Test To Show how THe RSS Feeder is Working
        public static void main(String[] args) throws SAXException, IOException, ParserConfigurationException {

                RSSFeedParser parser = new RSSFeedParser("http://rss.walla.co.il/?w=/6/0/12/@rss.e");
                Feed feed = parser.readFeed();
                System.out.println(feed);
                for (FeedMessage message : feed.getMessages()) {
                        System.out.println(message);

                }
        }

        // Check If rss (return true if it is)
        public static boolean isRSS(String URL) throws ParserConfigurationException, SAXException, IOException {
                DocumentBuilder builder = DocumentBuilderFactory.newInstance().newDocumentBuilder();
                Document doc = builder.parse(URL);
                return doc.getDocumentElement().getNodeName().equalsIgnoreCase("rss");
        }
 }
```

מחלקת FeedMessage:

```
public class FeedMessage {

    String title;
```

```
4.      String description;
5.      String link;
6.      String author;
7.      String guid;
8.      //Return The Title
9.      public String getTitle() {
10.         return title;
11.     }
12.     //Sets The title
13.     public void setTitle(String title) {
14.         this.title = title;
15.     }
16.     //Return the Description
17.     public String getDescription() {
18.         return description;
19.     }
20.     //Set the Description
21.     public void setDescription(String description) {
22.         this.description = description;
23.     }
24.     //Return the link
25.     public String getLink() {
26.         return link;
27.     }
28.     //Set the link
29.     public void setLink(String link) {
30.         this.link = link;
31.     }
32.     //Return the Author
33.     public String getAuthor() {
34.         return author;
35.     }
36.     //Set the Author
37.     public void setAuthor(String author) {
38.         this.author = author;
39.     }
40.     //Return the GUID
41.     public String getGuid() {
42.         return guid;
43.     }
44.     //Set the GUID
45.     public void setGuid(String guid) {
46.         this.guid = guid;
47.     }
48.     //ToString
49.     @Override
50.     public String toString() {
51.         return "FeedMessage [title=" + title + ", description=" +
    description
52.                 + ", link=" + link + ", author=" + author + ",
    guid=" + guid
53.                 + "]";
54.     }
```

}**מחלקת Feed:**

```
1. import java.util.ArrayList;
```

```
2. import java.util.List;
3.
4. public class Feed {
5.
6.     final String title;
7.     final String link;
8.     final String description;
9.     final String language;
10.      final String copyright;
11.      final String pubDate;
12.
13.      final List<FeedMessage> entries = new ArrayList<FeedMessage>();
14.      //Constructor
15.      public Feed(String title, String link, String description,
   String language,
16.              String copyright, String pubDate) {
17.          this.title = title;
18.          this.link = link;
19.          this.description = description;
20.          this.language = language;
21.          this.copyright = copyright;
22.          this.pubDate = pubDate;
23.      }
24.      //Return the messages
25.      public List<FeedMessage> getMessages() {
26.          return entries;
27.      }
28.      //Return the title
29.      public String getTitle() {
30.          return title;
31.      }
32.      //Return The link
33.      public String getLink() {
34.          return link;
35.      }
36.      //Return the Description
37.      public String getDescription() {
38.          return description;
39.      }
40.      //Return the language.
41.      public String getLanguage() {
42.          return language;
43.      }
44.      //Return the CopyRight
45.      public String getCopyright() {
46.          return copyright;
47.      }
48.      //Return the publishDate.
49.      public String getPubDate() {
50.          return pubDate;
51.      }
52.      //ToString
53.      @Override
54.      public String toString() {
55.          return "Feed [copyright=" + copyright + ", description=" +
   description
56.                  + ", language=" + language + ", link=" + link + ",
   pubDate="
57.                  + pubDate + ", title=" + title + "]";
58.      }
59.
```

```
60.  }
```

מחלקת Client:

```
1. import java.io.DataInputStream;
2. import java.io.DataOutputStream;
3. import java.io.IOException;
4. import java.net.Socket;
5.
6. class Client {
7.
8.  // IO streams
9.  private static DataOutputStream toServer;
10.   private static DataInputStream fromServer;
11.
12.   private Socket socket;
13.  //Constractor - Client
14.   public Client(){
15.
16.      try {
17.       // Creates a socket in order to connect to the server
18.       socket = new Socket("localhost", 8000);
19.
20.    // Create an input stream to receive data from the server
21.     fromServer = new DataInputStream(socket.getInputStream());
22.
23.     // Create an output stream to send data to the server
24.     toServer = new DataOutputStream(socket.getOutputStream());
25.           }
26.   catch (IOException ex) {   }
27.  }
28.  //Send the URL (message) to the server.
29.   public void writeToServer(String st){
30.     try {
31.          toServer.writeUTF(st);
32.          toServer.flush();
33.    } catch (IOException e) {e.printStackTrace(); }
34.   }
35.
36.  //Read the feed (messages) from the server.
37.   public String readFromServer(){
38.     try {
39.             return fromServer.readUTF();
40.    } catch (IOException e) {e.printStackTrace(); }
41.  return "Null";
42.  }
43.
44.  }
```

ביבליוגרפיה:

https://stackoverflow.com/ - עזרה עם הקוד ותשובות לבאגים ובעיות.

ספריות חיצוניות של הפרויקט:

SwingPlus 2.12

http://www.java2s.com/example/jar/s/swingplus-2.12-index.html