

***الحمد لله الذي جعل لنا من العلم نورا نهدي به و بعد
نتقدم ببحثنا هذا الى زملائنا التلاميذ و الى كل من يجمعنا بهم رباط العلم من مستمعين و قراء و مدرسين فهذا البحث يشمل .....الذي نامل ان يعجبكم
و اذا نحن نضع بين ايديكم هذا البحث الذي نرجو ان يكون في المستوى......***

 ***مقدمة عن جمع ومراقبة حركة مرور البيانات فى الشبكة :***

يمكن أن تكون البيانات مفيدة لأغراض العديد من بما في ذلك استكشاف الأخطاء وإصلاحها تحليل حركة مرور شبكة الاتصال و لأغراض الأمان.

تعتبر **مراقبة حركة المرور علي الشبكة** عملية قياس كم ونوع الحركة المرورية علي شبكة معينة. ويعتبر ذلك ضروريا بشكل خاص لإدارة نقل البيانات بشكل فعال.

**الادوات :**

تتوافر أدوات البرمجيات المتنوعة لقياس **حركة المرور على الشبكة** , تقيس بعض الأدوات حركة المرور عن طريق الالتقاط بينما تستخدم أخرى **بروتوكول ادارة الشبكات البسيط.** إن أدوات التحكم في الشبكة WMI وغيرها من العوامل الداخلية لقياس استخدام ناقل البيانات علي آلات أو موجهات فردية. لكن النوع الأخير لا يكتشف نوع الحركة المرورية ولا تعمل علي آلات لا تستخدم ال**عميل البرمجى** اللازم مثل الآلات الزائفة علي الشبكة أو الآلات التي لا يتوافر فيها عميل. في الحالة الأخيرة, يُفضل استخدام الأجهزة المباشرة. عادة ما يظهر ذلك بين الشبكة المحلية ونقطة الخروج من الشبكة المحلية, وبوجه عام فإن الشبكات الموسعة أو موجهات الإنترنت وكافة الحزم تخرج وتدخل إلي الشبكة عن طريقها. في معظم الحالات قد تعمل الأجهزة بمثابة جسر علي الشبكة فلا يكتشفها المستخدمين.

**الوظائف والسمات:**

**تتسم أدوات القياس عامة بالوظائف والسمات التالية:**

* واجهة المستخدم :الشبكة, الرسوم التصويرية، لوحة التحكم
* بيانات الحركة المرورية الفعلية
* وصف نشاط الشبكة في ضوء قواعد المطابقة المرورية المحددة مسبقا لإظهار:
	+ عنوان الإنترنت الخاص بالجهاز المستخدم
	+ عنوان الإنترنت الخاص بالجهاز عن بُعد
	+ رقم المنفذ أو [البروتوكول](http://ar.wikipedia.org/wiki/%D8%A7%D9%84%D8%A8%D8%B1%D9%88%D8%AA%D9%88%D9%83%D9%88%D9%84)
	+ اسم المستخدم المسجل
* مساحات نقل البيانات
* دعم تشكيل الحركة المرورية أو تحديد المعدلات (التداخل مع صفحة ضبط الحركة المرورية بالشبكة)
* دعم حجب الموقع وفلترة المحتوي
* إنذارات لإبلاغ المسؤول في حالة الاستخدام الزائد (سواء بالعنوان الخاص بالجهاز أو ككل)

### بعض الأدوات المتاحة:

**تتضمن بعض الأدوات المتاحة ما يلي:**

* حزم عمليات نظام توليد سجلات التدقيق والاستثمار في تدفق شبكي مفصل لبيانات العمليات, والأداء, وإدارة الأمان.
* برنامج (كاكتي Cacti) يسمح للمستخدم استخدام خدمات الاقتراع في فترات زمنية محددة مسبقا وعمل رسوم توضيحية بالبيانات الناتجة.
* برنامج تسريع الاتصال بالإنترنت cFosSpeed يقوم بتصنيف الحركة المرورية ويدع المستخدم يعرض ويُشكل ويُعرّف أو يستخدم بروتوكولات تحديد السرعة أو البرامج المثبتة في النوافذ.
* جهاز مراقبة التدفق FlowMon يعتبر الحل الكامل لمراقبة تدفق الحركة المرورية فس الشبكة وتحليلها بالإضافة إلي تشغيل مجمعات 10 جيجابت وغيرها من نظم الإشراف.
* برنامج InterMapper لتحديد مصادر الأعطال المُصنع خصيصاً لأجهزة ماكنتوش .
* يقدم برنامج LiveAction رؤية طبقة التوجيه الفعلي مما يسمح للمستخدم رؤية وحل مشكلة الموجهات وتنفيذ التوجيه القائم علي سياسة.
* راسم المرور متعدد المسار.
* محدد الشبكة NetLimiter برنامج لمراقبة المرور وتشكيل البرمجيات لبرامج الويندوز.
* برنامج OmniPeek أحد الحلول لمراقبة الشبكة من الخادم إلي الخادم مما يوفر الدعم للعديد من محولات الحزم والمجمعات عن بُعد.
* برنامج Observium وهو تطبيق أتوماتيكي لمراقبة الشبكة يركز علي جمع البيانات المكثف ورسم البنية التحتية للشبكة.
* برنامج بايسلار راسم مرور الموجه PRTG ويعمل علي برنامج الويندوز مع واجهات تصويرية وشبكية ويركز علي الحزم التي تستخدم برنامج سيسكو نت فلو والتقاط الحزم أو بروتوكول إدارة الشبكات البسيط لمراقبة استخدامات ناقل البيانات.
* شبكات PacketTrap ـ محلل للمرور ولتدفق المرور.
* المدقق Scrutinizer لتحليل تدفق الشبكة والتدفق المروري وييح الرؤية الواضحة للسلوك المروري بالشبكة وإتجاهاته. إن تحسين تحليل المرور لبيانات نت فلو وجا-فلو وإس-فلو يحدد أي من المستخدمين والتطبيقات تستهلك أي ناقل للبيانات.
* برنامج Sandvine أحد الحلول الذكية للشبكات لقياس وإدارة مرور الشبكة باستخدام سياسة مفاتيح المرور.
* نظام SevOne لمراقبة أداء الشبكة.

**مقدمة عن تحليل حركة مرور البيانات فى الشبكة :**

هو معرفة الباندوذ وهل هى كافية بالنسبة لمستخدم , تحليل حركة مرور ويب هو. بسيطة شرح وتحليل حركة مرور الشبكة هو معالجة البيانات الإحصائية فهم تصنيفات محرك البحث،يساعدة فى معرفة سرعة انتقال البيانات .

**برامج تحليل المرور:**

* [**BitAnalyzer**](http://ar.downv.com/download-BitAnalyzer-10086698.htm)**.**
* Esp
* Netjini
* **HTTPREPLAY**

**مقدمة فى التحكم بمرور البيانات فى الشبكة :**

هو اداره البيانات الخارجة والداخلة الى الشبكة والسيطره على مواردها وحمايتها من اى خطر .

**مزايا التحكم بالمرور:**

* المحافظة على النطاق الترددي عن طريق منع الوصول إلى مواقع الويب ذات حركة المرور العالية. هذا إلى جانب حماية شبكة الشركة من عمليات اختراق الأمان بجدار حماية ثنائي الاتجاه في وضع التعليم.
* والتحكم في الوسائط والأجهزة غير المصرح بها في الشبكة. مراقبة الأمان في الشبكة بالكامل من موقع واحد.
* إدراج مجموعة من خيارات التخصيص وإعداد التقارير من أجل تهيئة أمان النظام وفقًا لمتطلباتك.
* موازنة تحميل تنفيذ المهام لمنع التحميل الزائد على اتصالات الشبكة ومنع اقتحام برامج مكافحة الفيروسات في وقتٍ واحد لمضيفي الأجهزة الظاهرية.

**من برامج التحكم فى المرور:**

**DeskSoft BWMeter v6.3.0-1**

**برنامج رائع للتحكم بالشبكة و بحركة كل شاردة و واردة من والى الانترنت ومراقبة حركة المرور الداونلود والابلود

هو عرض النطاق الترددي ، ورصد و تحكم الحركة، والذي يقيس ، ويعرض و يسيطر على كل حركة المرور من والى جهاز الكمبيوتر أو على شبكة الاتصال . على عكس المنتجات الأخرى ، فإنه يمكن تحليل حزم البيانات ( من أين جاءت ، وأين تذهب ، و بروتوكول الذين يستخدمونها ) . هذا يجعل من الممكن التمييز بين حركة المرور المحلية وشبكة الإنترنت. ويمكن أيضا أن تستخدم الـ BWMeter لمراقبة حركة المرور من خلال تحديد الحد الأقصى للسرعة لجميع أنواع التوصيلات أو تقييد الوصول إلى بعض المواقع على شبكة الإنترنت . أنه يقوم بعمل إحصاءات لكافة أجهزة الكمبيوتر في الشبكة ، وقياس وعرض كافة حركة المرور LAN ، وكذلك التنزيل و التحميل من شبكة الانترنت . حتى انه يمكنك تحديد المرشحات التي تظهر النقل الخاص بك مع بعض عناوين الإنترنت (على سبيل المثال لمعرفة كمية البيانات التي تقوم بتحميلها من موقع الأخبار المفضل لديك) .**

***This article provides an introduction to libpcap, and shows, with examples of source code, how you can use it to create your own packet-sniffing programs.***

All data on the network travels in the form of packets, which is the data unit for the network. To understand the data a packet contains, we need to understand the protocol hierarchy in the reference models. I recommend that if you don’t know the ISO OSI (Open Systems Interconnection) Reference Model, you should read up on it. A good starting point is Wikipedia.

The network layer is where the term packet is used for the first time. Common protocols at this layer are IP (Internet Protocol), ICMP (Internet Control Message Protocol), IGMP (Internet Group Management Protocol) and IPsec (a protocol suite for securing IP). The transport layer’s protocols include TCP (Transmission Control Protocol), a connection-oriented protocol; UDP (User Datagram Protocol), a connection-less protocol; and SCTP (Stream Control Transmission Protocol), which has features of both TCP and UDP. The application layer has many protocols that are commonly used, like HTTP, FTP, IMAP, SMTP and more.

Capturing packets means collecting data being transmitted on the network. Every time a network card receives an Ethernet frame, it checks if its destination MAC address matches its own. If it does, it generates an interrupt request. The routine that handles this interrupt is the network card’s driver; it copies the data from the card buffer to kernel space, then checks the ethertype field of the Ethernet header to determine the type of the packet, and passes it to the appropriate handler in the protocol stack. The data is passed up the layers until it reaches the user-space application, which consumes it.

When we are sniffing packets, the network driver also sends a copy of each received packet to the packet filter. To sniff packets, we will use libpcap, an open source library.

***فهم (libpcap):-***

يببكاب هي مكتبة مفتوحة المصدر منصة مستقلة لالتقاط الحزم (نسخة ويندوز هو winpcap). المتشممون الشهيرة مثل TCPDUMP ويريشارك الاستفادة من هذه المكتبة.

لكتابة برنامج التقاط حزمة لدينا، ونحن في حاجة الى واجهة الشبكة التي للاستماع. يمكننا تحديد هذا الجهاز، أو استخدام وظيفة التي تنص يببكاب: char \*pcap\_lookupdev(char \*errbuf) .

هذا بإرجاع مؤشر إلى سلسلة تحتوي على اسم الجهاز أول شبكة مناسبة لالتقاط حزم؛ على خطأ، فإنها ترجع خالي (مثل وظائف يببكاب الأخرى). و errbuf هو عازلة الموفر من قبل المستخدم لتخزين رسالة خطأ في حالة وجود خطأ - أنه مفيد جدا لتصحيح أخطاء البرنامج الخاص بك. يجب أن يكون هذا المخزن المؤقت قادرة على الاستمرار على الأقل PCAP\_ERRBUF\_SIZE (حاليا 256) بايت.

**Getting control of the Network Device:**

Next, we open the chosen network device using the function pcap\_t \*pcap\_open\_live(const char \*device, int snaplen, int promisc, int to\_ms, char \*errbuf). It returns an interface handler of type pcap\_t, which other libpcap functions will use. The first argument is the network interface we want to open; the second is the maximum number of bytes to capture.

Setting it to a low value will be useful when we only want to grab packet headers. The Ethernet frame size is 1518 bytes. A value of 65535 will be enough to hold any packet from any network. The promisc flag indicates whether the network interface should be put into promiscuous mode or not. (In promiscuous mode, the NIC will pass all frames it receives to the CPU, instead of just those addressed to the NIC’s MAC address. Read more on Wikipedia(.

The to\_ms option tells the kernel to wait for a particular number of milliseconds before copying information from kernel space to user space. A value of zero will cause the read operation to wait until enough packets are collected. To save extra overhead in copying from kernel space to user space, we set this value according to the volume of network traffic.

**القبض الفعلي:**

Actual capture:

نحن بحاجة إلى البدء في الحصول على الحزم. دعونا استخدام u\_char \*pcap\_next(pcap\_t \*p, struct pcap\_pkthdr \*h) . هنا، \*p هو مؤشر إرجاعها بواسطة pcap\_open\_live(); حجة أخرى هو مؤشر إلى متغير من نوع struct pcap\_pkthdr التي يتم إرجاعها الحزمة الأولى التي يصل.

وظيفة int pcap\_loop(pcap\_t \*p, int cnt, pcap\_handler callback, u\_char \*user)يستخدم لجمع الحزم ومعالجتها. فإنه سيعود عندما cnt تم القبض على عدد من الحزم. ويتم استخدام وظيفة رد الاتصال لمعالجة الحزم القبض على (نحن بحاجة لتعريف هذه الدالة رد الاتصال). لتمرير معلومات اضافية لهذه الوظيفة، ونحن نستخدم \*user المعلمة، وهو مؤشر إلى u\_char متغير (سيتعين علينا أن يلقي ذلك بأنفسنا، وفقا لاحتياجاتنا في وظيفة رد الاتصال).

يجب أن يكون رد وظيفة التوقيع على شكل: void callback\_function(u\_char \*arg, const struct pcap\_pkthdr\* pkthdr, const u\_char\* packet) . الوسيطة الأولى هي \*user parameter مررنا إلىpcap\_loop() ؛ الحجة المقبل هو مؤشر إلى الهيكل الذي يحتوي على معلومات حول حزمة القبض عليه. هيكل struct pcap\_pkthdr هو على النحو التالي (من pcap.h ):

struct pcap\_pkthdr {

        struct timeval ts;   /\* time stamp \*/

        bpf\_u\_int32 caplen;  /\* length of portion present \*/

        bpf\_u\_int32 len;     /\* length of this packet (off wire) \*/

};

An alternative to pcap\_loop() is pcap\_dispatch(pcap\_t \*p, int cnt, pcap\_handler callback, u\_char \*user). The only difference is that it returns when the timeout specified inpcap\_open\_live() is exceeded.

**تحليل و تصفية حركة المرور:**

Filtering traffic:

لقد تم فقط الحصول على جميع الحزم القادمة إلى الواجهة. الآن، سنستخدم pcap وظيفة التي تسمح لنا لتحديد حركة المرور القادمة إلى منفذ معين. ونحن قد تستخدم هذا فقط الحزم عملية بروتوكول معين، مثل ARP أو المرور FTP، على سبيل المثال. أولا، لدينا لتجميع التصفية باستخدام التعليمات البرمجية التالية:

int pcap\_compile(pcap\_t \*p, struct bpf\_program \*fp, const char \*str, int optimize, bpf\_u\_int32 mask);

The first argument is the same as before; the second is a pointer that will store the compiled version of the filter. The next is the expression for the filter. This expression can be a protocol name like ARP, IP, TCP, UDP, etc. You can see a lot of sample expressions in the pcap-filter or tcpdump man pages, which should be installed on your system.

The next argument indicates whether to optimise or not (0 is false, 1 is true). Then comes the netmask of the network the filter applies to. The function returns -1 on error (if it detects an error in the expression).

الحجة المقبل إلى ما إذا كان لتحسين أو لا (0 باطل، 1 صحيح). ثم يأتي قناع الشبكة ينطبق فلتر ل. تقوم الدالة بإرجاع -1 على الخطأ (إذا كشف خطأ في التعبير).

بعد تجميع، دعونا تطبيق عامل التصفية باستخدام int pcap\_setfilter(pcap\_t \*p, struct bpf\_program \*fp) .

**Finding IPv4 information:**

int pcap\_lookupnet(const char \*device, bpf\_u\_int32 \*netp, bpf\_u\_int32 \*maskp, char \*errbuf).

نستخدم هذه الوظيفة للعثور على عنوان الشبكة وقناع الشبكة IPv4 و المرتبطة مع الجهاز. سوف تعاد العنوان في \*netp والقناع في \*mask .

**A small sniffer program:**

Now let’s write a small sniffer program that will help us understand how pcap works.

#include <pcap.h>

#include <stdio.h>

#include <stdlib.h>

#include <errno.h>

#include <sys/socket.h>

#include <netinet/in.h>

#include <arpa/inet.h>

#include <netinet/if\_ether.h>

إرسال وظيفة رد الاتصال لمعالجة الحزم لدينا القبض عليه. هذه الوظيفة يطبع للتو تشغيل عدد من الحزم، والقبض عليه. بعد ذلك سنقوم إرسال ظيفة رد آخر

void my\_callback(u\_char \*args, const struct pcap\_pkthdr\* pkthdr, const u\_char\*

    packet)

{

    static int count = 1;

    fprintf(stdout, "%3d, ", count);

    fflush(stdout);

    count++;

}

Now comes the main() function. We can make use of the functions we learnt about earlier, in this function:

int main(int argc,char \*\*argv)

{

    int i;

    char \*dev;

    char errbuf[PCAP\_ERRBUF\_SIZE];

    pcap\_t\* descr;

    const u\_char \*packet;

    struct pcap\_pkthdr hdr;

    struct ether\_header \*eptr;    /\* net/ethernet.h \*/

    struct bpf\_program fp;        /\* hold compiled program \*/

    bpf\_u\_int32 maskp;            /\* subnet mask \*/

    bpf\_u\_int32 netp;             /\* ip \*/

    if(argc != 2){

        fprintf(stdout, "Usage: %s \"expression\"\n"

            ,argv[0]);

        return 0;

    }

    /\* Now get a device \*/

    dev = pcap\_lookupdev(errbuf);

    if(dev == NULL) {

        fprintf(stderr, "%s\n", errbuf);

        exit(1);

    }

        /\* Get the network address and mask \*/

    pcap\_lookupnet(dev, &netp, &maskp, errbuf);

    /\* open device for reading in promiscuous mode \*/

    descr = pcap\_open\_live(dev, BUFSIZ, 1,-1, errbuf);

    if(descr == NULL) {

        printf("pcap\_open\_live(): %s\n", errbuf);

        exit(1);

    }

    /\* Now we'll compile the filter expression\*/

    if(pcap\_compile(descr, &fp, argv[1], 0, netp) == -1) {

        fprintf(stderr, "Error calling pcap\_compile\n");

        exit(1);

    }

    /\* set the filter \*/

    if(pcap\_setfilter(descr, &fp) == -1) {

        fprintf(stderr, "Error setting filter\n");

        exit(1);

    }

    /\* loop for callback function \*/

    pcap\_loop(descr, -1, my\_callback, NULL);

    return 0;

}

run it as the root (necessary for permissions to execute in promiscuous mode):

$ gcc -lpcap sniff.c -o sniffer

# ./sniffer ip

**المخرج يكون فى هذا الشكل :**



قدمنا ip كوسيلة للتعبير، وملء الشاشة قريبا مع عدد من عدد من الحزم IP. يمكنك استبدال ip مع أي تعبير من اختيارك، مثل برنامج التعاون الفني او اى شئ اخر .

Here’s another callback function, which will display the contents of the packets accepted by your filter expression (it’s already in sniff.c):

void another\_callback(u\_char \*arg, const struct pcap\_pkthdr\* pkthdr,

        const u\_char\* packet)

{

    int i=0;

    static int count=0;

    printf("Packet Count: %d\n", ++count);    /\* Number of Packets \*/

    printf("Recieved Packet Size: %d\n", pkthdr->len);    /\* Length of header \*/

    printf("Payload:\n");                     /\* And now the data \*/

    for(i=0;i<pkthdr->len;i++) {

        if(isprint(packet[i]))                /\* Check if the packet data is printable \*/

            printf("%c ",packet[i]);          /\* Print it \*/

        else

            printf(" . ",packet[i]);          /\* If not print a . \*/

        if((i%16==0 && i!=0) || i==pkthdr->len-1)

            printf("\n");

    }

}

قمنا بتعديل pcap\_loop() خط في main() ، الذي يدعو my\_callback() ، استدعاء هذه الدالة رد الاتصال بدلا من ذلك.ترجمة البرنامج تم تغييرها، وتشغيله مع نفس التعبير كما حجتها.

وهنا المخرجات عباره عن محتويات ال :ip



كود تحليل بلغة الجافا:

import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.Container;

import java.awt.HeadlessException;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JApplet;

import javax.swing.JButton;

/\*

 \* Created on Oct 6, 2004

 \*

 \* @author Steve Tanimoto

 \*/

public class TrafficLight extends JApplet implements ActionListener {

 int phase = 0;

 Container c;

 /\*\*

 \* @throws java.awt.HeadlessException

 \*/

 public TrafficLight() throws HeadlessException {

 JButton jb = new JButton("Change!");

 c = this.getContentPane();

 c.setLayout(new BorderLayout());

 c.add(jb, BorderLayout.SOUTH);

 jb.addActionListener(this);

 c.setBackground(Color.green);

 }

 public void actionPerformed(ActionEvent e) {

 phase = (phase + 1) % 3;

 if (phase == 0) {

 c.setBackground(Color.green);

 }

 else if (phase == 1) {

 c.setBackground(Color.yellow);

 }

 else if (phase == 2) {

 c.setBackground(Color.red);

 }

 }

}

***وهكذا ينتهى الجزء التحليلى من المشروع ونأتى الى الجزء الثانى منه وهو المراقبه :***

I have done this far...( look at attachment)

|  |
| --- |
| // Three traffic lights shown in a panel |
| 02 |   class Light |

|  |  |
| --- | --- |
| 03 |       extends JPanel { |
| 04 |     private boolean red; |

|  |  |
| --- | --- |
| 05 |     private boolean yellow; |
| 06 |     private boolean green; |

|  |  |
| --- | --- |
| 07 |   |
| 08 |     /\*\* Default constructor \*/ |

|  |  |
| --- | --- |
| 09 |     public Light() { |
| 10 |     } |

|  |  |
| --- | --- |
| 11 |   |
| 12 |     /\*\* Set red light on \*/ |

|  |  |
| --- | --- |
| 13 |     public void turnOnRed() { |
| 14 |       red = true; |

|  |  |
| --- | --- |
| 15 |       yellow = false; |
| 16 |       green = false; |

|  |  |
| --- | --- |
| 17 |       repaint(); |
| 18 |     } |

|  |  |
| --- | --- |
| 19 |   |
| 20 |     /\*\* Set yellow light on \*/ |

|  |  |
| --- | --- |
| 21 |     public void turnOnYellow() { |
| 22 |       red = false; |

|  |  |
| --- | --- |
| 23 |       yellow = true; |
| 24 |       green = false; |

|  |  |
| --- | --- |
| 25 |       repaint(); |
| 26 |     } |

|  |  |
| --- | --- |
| 27 |   |
| 28 |     /\*\* Set green light on \*/ |

|  |  |
| --- | --- |
| 29 |     public void turnOnGreen() { |
| 30 |       red = false; |

|  |  |
| --- | --- |
| 31 |       yellow = false; |
| 32 |       green = true; |

|  |  |
| --- | --- |
| 33 |       repaint(); |
| 34 |     } |

|  |  |
| --- | --- |
| 35 |   |
| 36 |     /\*\* Display lights \*/ |

|  |  |
| --- | --- |
| 37 |     protected void paintComponent(Graphics g) { |
| 38 |       super.paintComponent(g); |

|  |  |
| --- | --- |
| 39 |   |
| 40 |       if (red) { |

|  |  |
| --- | --- |
| 41 |         g.setColor(Color.red); |
| 42 |         g.fillOval(10, 10, 20, 20); |

|  |  |
| --- | --- |
| 43 |         g.setColor(Color.black); |
| 44 |         g.drawOval(10, 35, 20, 20); |

|  |  |
| --- | --- |
| 45 |         g.drawOval(10, 60, 20, 20); |
| 46 |         g.drawRect(5, 5, 30, 80); |

|  |  |
| --- | --- |
| 47 |       } |
| 48 |       else if (yellow) { |

|  |  |
| --- | --- |
| 49 |         g.setColor(Color.yellow); |
| 50 |         g.fillOval(10, 35, 20, 20); |

|  |  |
| --- | --- |
| 51 |         g.setColor(Color.black); |
| 52 |         g.drawRect(5, 5, 30, 80); |

|  |  |
| --- | --- |
| 53 |         g.drawOval(10, 10, 20, 20); |
| 54 |         g.drawOval(10, 60, 20, 20); |

|  |  |
| --- | --- |
| 55 |       } |
| 56 |       else if (green) { |

|  |  |
| --- | --- |
| 57 |         g.setColor(Color.green); |
| 58 |         g.fillOval(10, 60, 20, 20); |

|  |  |
| --- | --- |
| 59 |         g.setColor(Color.black); |
| 60 |         g.drawRect(5, 5, 30, 80); |

|  |  |
| --- | --- |
| 61 |         g.drawOval(10, 10, 20, 20); |
| 62 |         g.drawOval(10, 35, 20, 20); |

|  |  |
| --- | --- |
| 63 |       } |
| 64 |       else { |

|  |  |
| --- | --- |
| 65 |         g.setColor(Color.black); |
| 66 |         g.drawRect(5, 5, 30, 80); |

|  |  |
| --- | --- |
| 67 |         g.drawOval(10, 10, 20, 20); |
| 68 |         g.drawOval(10, 35, 20, 20); |

|  |  |
| --- | --- |
| 69 |         g.drawOval(10, 60, 20, 20); |
| 70 |       } |

|  |  |
| --- | --- |
| 71 |     } |
| 72 |   |

|  |  |
| --- | --- |
| 73 |     /\*\* Set preferred size \*/ |
| 74 |     public Dimension getPreferredSize() { |

|  |  |
| --- | --- |
| 75 |       return new Dimension(40, 90); |
| 76 |     } |

|  |  |
| --- | --- |
| 77 |   }//end of class |

***اكواد لتحكم بالحزم:***

import java.util.\*;

public class Main {

static Scanner in = new Scanner(System.in);

public static void main(String[] args) {

 do {

 try {

 System.out.println("Please enter command to proceed: ");

 System.out.println("Enter new aircraft = e");

 System.out.println("Display all aircraft = d");

 System.out.println("Show specific flight = s");

 System.out.println("Remove specific flight = r");

 String command = in.next();

 in.next(command);

 if ( (in.next(command)).equals("e") ) {

 ATControl.addToList(); // need to somehow "start" this class

 } else if ( (in.next(command)).equals("d") ) {

 ATControl.displayAll();

 } else if ( (in.next(command)).equals("s") ){

 ATControl.showFlight();

 } else if ( (in.next(command)).equals("r") ) {

 ATControl.removeFlight();

 } else if ( (in.next(command)).equals(null) ) {

 }

 } catch (InputMismatchException exc) {

 System.out.println("Wrong entry, please try again:");

 }

 } while (true);

}

}

***Linked List and Node - I called it Aircraft. I think this is where the list is stored and created. Manipulation to the list occurs in the next class (ATControl), or at least I think it will.***

package airTraffic;

import java.util.LinkedList;

public class Aircraft {

// stores data

private static final int INITIAL\_ALLOCATION = 20;

private int size = INITIAL\_ALLOCATION;

//declare LinkedList and node names

static LinkedList <String> list = new LinkedList <String> ();

private Aircraft head = new Aircraft ();

private Aircraft tail = new Aircraft ();

// tells list to add nodes

public void addNodes (int n, LinkedList<String> s) {

 s = list;

 head.next = tail;

 tail.next = tail;

 size = n;

 Aircraft temp = head;

 for (int i= 0; i < size; ++i) {

 temp.next = new Aircraft ();

 temp = temp.next;

 }

 temp.next = tail;

}

private String value;

Aircraft craft;

public Aircraft (String v) {

 value = v;

}

public Aircraft () {

}

public String get () {

 return value;

}

public void set (String v) {

 value = v;

}

public Aircraft next = null;

//auto generated method from ATControl

public static void add(String flight) {

 // a for or while loop might be needed here. Seems to easy to just have an empty add class

}

//auto generated method from ATControl

public static void remove() {

}

 }

***ATControl class. This is where (I think) the list is manipulated, allowing the user to add, remove and show the flights.***

package airTraffic;

import java.util.\*;

public class ATControl{

// implement Aircraft class (node) - empty argument list??

Aircraft aircraft = new Aircraft ();

static Scanner in = new Scanner (System.in);

// list of planes

static String [] planeList = {"Wide-body Airliner = w", "Regional Airliner = r", "Private Plane = p",

 "Military = m", "Cargo only: c", "Unknown = u"};

//add plane and details

public static void addToList () {

 System.out.printf("Enter flight number: ");

 String flight = in.nextLine();

 Aircraft.add(flight);

 //type of plane

 System.out.printf("Enter type of plane, ", "Choose from: " + planeList);

 String type = in.nextLine();

 try {

 if (type == "w") {

 System.out.println("Wide-body Airliner");

 }else if (type == "r") {

 System.out.println("Regional Airliner");

 }else if (type == "p") {

 System.out.println("Private Plane");

 }else if (type == "m") {

 System.out.println("Military");

 }else if (type == "c") {

 System.out.println("Cargo only");

 }else if (type == "u") {

 System.out.println("Unknown");

 } else type = null;

 }

 catch (InputMismatchException i) {

 System.out.println("You must enter valid command: " + planeList);

 }

 Aircraft.add(type);

 //plane speed

 System.out.printf("Enter current speed: ");

 String speed = in.nextLine();

 Aircraft.add(speed);

 //add Altitude

 System.out.printf("Enter current altitude: ");

 String alt = in.nextLine();

 Aircraft.add(alt);

}

//show flight

public static void showFlight () {

 System.out.printf("Enter flight number for details: ");

 in.nextLine();

 Aircraft.get(Aircraft, index);

}

// display all flights

public static void displayAll () {

 System.out.printf("All flights: " );

}

//remove flight

public static void removeFlight () {

 System.out.printf("Enter flight number to be removed: ");

 in.nextLine();

 Aircraft.remove();

}

}

فى هذا المشروع الذى قومنا استخدمنا اكواد بلغة ال c++ &&java

ملاحظه :

فى الجزاء حق المراقبة ما قدرنا نشتغل عملى لانو الجافا ما بتشتغل فى اى جهاز من اجهزتنا .

***وختاما :-***

 **بدأ سن القلم أن ينكسر ... وتتمزق حواف الأوراق ...
والرياح تأخذ ماتبقى في أيدينا من لحظة جمعتنا فيها جل أنواع الإخاء ...
تلاقينا ... تفرقنا ... عدنا ... ثم ها نحن نطرق أبواب الرحيل ... لنصل إلى مفترق الطريق ... ونقطة النهاية ...
نسال الله أن يجمعنا بكم أياما عديدة وأزمنة مديدة ... فلتلهج الألسن بالدعاء ... ولتنطق الحروف بالرجاء ...
إن لم يكن لنا هنا لقاء ...
فللأخرى لنا رجاء وأمل .**