

FUNCIONAMENT BÀSIC DEL PROGRAMA EN JAVA

En aquest apartat s'explicarà bàsicament com funciona el programa, el llenguatge d'aquest, la seva estructura i algunes de les comandes que realitza. No s'explicaràn les ordres concretament, sinó que es farà referència al significat d'aquests.

El programa consta d'un WebServer o servidor de la xarxa i un Request o suplicador. La funció del primer és la d'oferir el servei de connexió entre els ordenadors i el segon ens permetrà d'executar una senzilla comanda.

El programa ja ha estat elaborat, el que farem serà analitzar-lo per tal d'entendre'l una mica.

Java és una tecnologia de desenvolupament que conté un kit de desenvolupament. El primer que farem, es descarregar-nos la versió gratuïta d'internet, a ser possible la versió que s'esmenta a l'apartat de metodologia.

Començarem amb l'arxiu WebServer, que és l'important i el que analitzarem. A mesura que anem trobant expressions noves, les explicarem detingudament:

```
import java.net.*;
import java.awt.*;
import java.io.*;
import java.util.*;
import java.awt.event.*;

class WebServer extends Thread {
private ServerSocket ssMain;
private String webRoot;
char ff[]=new char[63];
boolean fi=false;
public WebServer() {this(".");}
public WebServer(String webRoot){this(webRoot, 80);}
public WebServer(String webRoot,int portNo){
this.webRoot = webRoot;
try {ssMain = new ServerSocket(portNo);}catch(IOException e)
{System.out.println("IOException: " + e.getMessage());}
}

public void run() {
try {while(!fi){Socket socket = ssMain.accept(); new
Request(webRoot,socket,this).start();}}
catch(IOException e) {System.out.println("IOException: " +
e.getMessage());}
}

public static void main(String args[]) {
webServer ws = null;
if (args.length == 2)ws = new WebServer(args[0],
Integer.parseInt(args[1]));
else if (args.length == 1)ws = new WebServer(args[0]);
else if (args.length == 0)ws = new WebServer();
if (ws != null)ws.start();
else System.out.println("Error");}
}
```

Aquest és l'arxiu que analitzarem. Aquest està escrit en gramàtica Java i és el codi font.

Quan utilitzem la gramàtica de Java s'han de tenir en compte les següents normes:

- Comentaris: són introduïts pel programador per explicar o aclarir possibles dubtes en la programació.

- Sentències: una sentència és una línia de programa.

- Bloc de codi: es un grup de sentències que formen una unitat. Les sentències es poden agrupar en blocs per a que una sola pugui controlar de forma relativament senzilla l'execució de moltes altres. Els blocs de codi Java estan delimitats per les claus ({}).

- Estructura de fitxer: els components d'un fitxer de codi font en Java i l'ordre amb que s'han definit.

- Paraules reservades: paraules que s'han predefinit al llenguatge Java (no es poden usar com a identificadors).

- Identificadors: els noms que es donen a les classes, variables i funcions. Els identificadors tenen restriccions amb els seus caràcters. Existeixen alguns convenis opcionals de gran abast per als identificadors.

- Literals: valors constants que s'escriuen de manera diferent, depenent del tipus de dades que siguin: per exemple, per a diferenciar els caràcters "123" del número 123.

- Expressions: combinació de paraules que s'evaluen a un únic valor. Això vol dir que quan es compilen resulten una sola operació o expressió.

- Operadors: els operadors efectuen la suma, resta, multiplicació i altres funcions matemàtiques i no matemàtiques.

Un cop tinguem en compte aquests aspectes fonamentals de la gramàtica de Java ja podrem continuar realitzant l'anàlisi bàsic.

Començarem per les expressions següents:

```
import java.net.*;
import java.awt.*;
import java.io.*;
import java.util.*;
import java.awt.event.*;
```

Aquestes són anomenades sentències import. Són sentències predeterminades en el nucli API de Java.

```
class webServer extends Thread {
```

A continuació ens trobem la classe WebServer, una classe és una descripció d'un tipus especial d'objecte Java, incloent les instruccions particulars que no el formen. Aquesta rebrà el nom del document sempre.

Ens diu que conté el Thread (fil, traça d'execució):

```
private ServerSocket ssMain;
private String webRoot;
char ff[]=new char[63];
boolean fi=false;
public webServer() {this(".");}
public webServer(String webRoot){this(webRoot, 80);}
public webServer(String webRoot,int portNo){
this.webRoot = webRoot;
try {ssMain = new ServerSocket(portNo);}catch(IOException e)
{System.out.println("IOException: " + e.getMessage());}
}

public void run() {
try {while(!fi){Socket socket = ssMain.accept(); new
Request(webRoot,socket,this).start();}}
catch(IOException e) {System.out.println("IOException: " +
e.getMessage());}
}

public static void main(String args[]) {
webServer ws = null;
if (args.length == 2)ws = new webServer(args[0],
Integer.parseInt(args[1]));
```

```

else if (args.length == 1)ws = new webServer(args[0]);
else if (args.length == 0)ws = new webServer();
if (ws != null)ws.start();
else System.out.println("Error");}
}

```

En la primera traça trobem:

```

private ServerSocket ssMain;
private String webRoot;
char ff[]=new char[63];
boolean fi=false;

```

private és un modificador d'accessibilitat, que restringeix l'accés a una subclasse.

Char és una paraula reservada de Java.

Boolean és una variable que conté valors true o false.

Això ens indica que el ServerSocket escolta la xarxa per nosaltres. A més, ens diu que webRoot és el directori d'on nosaltres agafem els arxius.

webRoot es tracta d'un servidor directori on s'allotja la xarxa.

PortNo és el port on està escoltant el servidor (ServerSocket).

```

public webServer() {this(".");}
public webServer(String webRoot){this(webRoot, 80);}
public webServer(String webRoot,int portNo){
this.webRoot = webRoot;
try {ssMain = new ServerSocket(portNo);}catch(IOException e)
{System.out.println("IOException: " + e.getMessage());}
}

```

Aquí veiem una altre seguit de sentències.

Public és un modificador d'accessibilitat, que permet accedir a una subclasse. This es una paraula reservada per a formar una expressió, com que es vol referir a una instància actual de la classe en la que s'escriu el codi, l'utilitza per assignar el ".".

La paraula String nomena a una seqüència de caràcters. El tipus de dades String es en realitat un classe del nucli API. La classe String té una importància especial al Java, ja que el compilador reconeix les constants String. Els caràcters entre dos corxeres són reconeguts pel compilador com literals de tipus String.

Int es una declaració de variables.

Try i catch són paraules reservades de Java.

Java també conté operadors matemàtics i lògics, en aquest cas fem servir el signe = per verificar una igualtat.

```
public void run() {
    try {while(!fi){Socket socket = ssMain.accept(); new
    Request(webRoot,socket,this).start();}}
    catch(IOException e) {System.out.println("IOException: " +
    e.getMessage());}
}
```

Aquestes sentències generen un procés principal de bucle, generen constantment una pregunta al servidor per veure si té alguna ordre que rebre.

Void és una paraula reservada de Java.

```
public static void main(String args[]) {
    webServer ws = null;
    if (args.length == 2)ws = new webServer(args[0],
    Integer.parseInt(args[1]));
    else if (args.length == 1)ws = new webServer(args[0]);
    else if (args.length == 0)ws = new webServer();
    if (ws != null)ws.start();
    else System.out.println("Error");}
}
```

El nou signe == vol dir que verifiqui la igualtat.

Else i if són paraules reservades de Java.

Static és cada instància d'una classe que posseeix la seva pròpia copia de qualsevol variable membre. S'utilitzen per guardar informació global de seguiment sobre les instàncies d'una classe.

Aquestes altres sentències indiquen que si no es compleix la sintaxis correcta es mostri un missatge "Error".

Amb aquest últim bloc s'acaba la thread principal i única.