UNIVERSITÀ DI PISA

Laboratorio di reti

Assignments Correction

Prof. Laura Ricci (ricci@unipi.it)

Assistant: Andrea De Salve (desalve@unipi.it)

19-04-2016

2015/2016

Assignments

- Alarm Clock
- Pi Greco
- Ufficio Postale
- Gestione Auto
- WebLookUp
- Calcolatrice TCP
- Mini FTP
- Query Flickr

……

➔ Cover different topics

Java Concurrent | Java Basic I/O | Java Networking

Assignments

- Alarm Clock
- Pi Greco
- Ufficio Postale
- Gestione Auto

- WebLookUp
- Calcolatrice TCP
- Mini FTP
- Query Flickr

……

➔ Cover different topics

Java Concurrent | Java Basic I/O | Java Networking

Query Flickr

Flickr URL API https://api.flickr.com/services/feeds/photos_public.gne?format=json&nojsoncallback=1&tags=_____

Retrieve JSON FEEDS


```
{
        "title": "Recent Uploads tagged nature",
        "link": "https://www.flickr.com/photos/tags/nature/",
        "description": "",
        "modified": "2016-04-18T20:15:43Z",
        "generator": "https://www.flickr.com/",
        "items": [
    {
            "title": "Eurydema ventralis",
            "link": "https://www.flickr.com/photos/126744915@N05/25905146074/",
            "media": {"m":"https://farm2.staticflickr.com/1657/25905146074_71da10124d_m.jpg"},
            "date_taken": "2016-04-17T14:16:28-08:00",
            "description": " <p><a href=\"https://www.flickr.com/people/126744915@N05/\">gökhan er
/126744915@N05/25905146074/\" title=\"Eurydema ventralis\"><img src=\"https://farm2.staticflickr.com/1657/259(
ventralis\" /><\/a><\/p> <p><\/p>",
            "published": "2016-04-18T20:15:43Z",
            "author": "nobody@flickr.com (gökhan eren)",
            "author_id": "126744915@N05",
            "tags": "love nature türkiye beetle bugs böcek 2016 eurydemaventralis gkhnrn gökhanere
    },
    {
            "title": "Koenigsegg Agera R",
            "link": "https://www.flickr.com/photos/39464223@N08/26417662172/",
            "media": {"m":"https://farm2.staticflickr.com/1577/26417662172_0818468e6f_m.jpg"},
            "date_taken": "2014-06-13T19:19:03-08:00",
            "description": " <p><a href=\"https://www.flickr.com/people/39464223@N08/\">Nash FRoss
/39464223@N08/26417662172/\" title=\"Koenigsegg Agera R\"><img src=\"https://farm2.staticflickr.com/1577/26417
R\" /><\/a><\/p> ",
            "published": "2016-04-18T20:17:11Z",
            "author": "nobody@flickr.com (Nash FRosso)",
            "author_id": "39464223@N08",
            "tags": "sunset france slr nature beautiful car sport canon nice nikon couple gorgeous
le mclaren r rolls spotted gt photoshot lamborghini rs francia extérieur luxury b7 supercar bentley zonda koer
```


Download K images

. . .

Creating a Runnable object (1)

- A **Runnable** object for downloading images:

```
public class GetImage implements Runnable{

    private static int id=1;
    private String link;
    private int imageId;

    public GetImage(String linkImage){
        link=linkImage;
        imageId=GetImage.id++;
    }

@Override
public void run() {
    try {
        URL urlImage=new URL(link);
        URLConnection uc=urlImage.openConnection();
        BufferedInputStream oos=new BufferedInputStream(uc.getInputStream());
        File f=new File("Image"+imageId+".jpeg");
        BufferedOutputStream bos=new BufferedOutputStream(new FileOutputStream(f));
        int v;
        while((v=oos.read())!=-1){
            bos.write(v);
```

Creating a Runnable object (2)

- A **Runnable** object for downloading images:

```
public class GetImage implements Runnable{

    private static int id=1;
    private String link;
    private int imageId;

    public GetImage(String linkImage){
        link=linkImage;
        imageId=GetImage.id++;
    }

@Override
public void run() {
    try {
        URL urlImage=new URL(link);
        URLConnection uc=urlImage.openConnection();
        BufferedInputStream oos=new BufferedInputStream(uc.getInputStream());
        File f=new File("Image"+imageId+".jpeg");
        BufferedOutputStream bos=new BufferedOutputStream(new FileOutputStream(f));
        int v;
        while((v=oos.read())!=-1){
            bos.write(v);
```

connect not required!

Creating a Runnable object (3)

- A **Runnable** object for downloading images:

```
public class GetImage implements Runnable{

    private static int id=1;
    private String link;
    private int imageId;

    public GetImage(String linkImage){
        link=linkImage;
        imageId=GetImage.id++;
    }

@Override
public void run() {
    try {
        URL urlImage=new URL(link);
        URLConnection uc=urlImage.openConnection();
        BufferedInputStream oos=new BufferedInputStream(uc.getInputStream());
        File f=new File("Image"+imageId+".jpeg");
        BufferedOutputStream bos=new BufferedOutputStream(new FileOutputStream(f));
        int v;
        while((v=oos.read())!=-1){
            bos.write(v);
```

Operations that depend on being connected, like getInputStream, getOutputStream, etc, will implicitly perform the connection, if necessary.

The Main: QueryFliker (1)

```
public class Main {

    static final String pathName="https://api.flickr.com/services/feeds/photos_public
    //Tag
    private static final String tags="&tags=nature";
    //Port
    static final int port=443;
    //Images Number
    static final int numeroRisultati=3;

    public static void main(String[] args){
        ThreadPoolExecutor pool = (ThreadPoolExecutor) Executors.newCachedThreadPool();
        try {
            //Mi collego alla URL
            URL url=new URL(pathName+tags);
            URLConnection uc=url.openConnection();
            uc.connect();
            System.out.println("Connecting to Flickr....");
```

The Main: QueryFliker (2)

```
public class Main {

    static final String pathName="https://api.flickr.com/services/feeds/photos_public
    //Tag
    private static final String tags="&tags=nature";
    //Port
    static final int port=443;
    //Images Number
    static final int numeroRisultati=3;

    public static void main(String[] args){
        ThreadPoolExecutor pool = (ThreadPoolExecutor) Executors.newCachedThreadPool();
        try {
            //Mi collego alla URL
            URL url=new URL(pathName+tags);
            URLConnection uc=url.openConnection();
            uc.connect();
            System.out.println("Connecting to Flickr....");
```

Create URL

Connect

The Main: QueryFliker (3)

```
//Leggo JSON
BufferedReader in=new BufferedReader(new InputStreamReader(uc.getInputStream()));
String line=null;
StringBuffer sb=new StringBuffer();
while((line=in.readLine())!=null){
    System.out.println(line);
    sb.append(line);
}
//Parse JSON
JSONObject obj = (JSONObject) new JSONParser().parse(sb.toString());
JSONArray results = (JSONArray) obj.get("items");
//Leggo i primi K link
if(numeroRisultati<=results.size()){
```

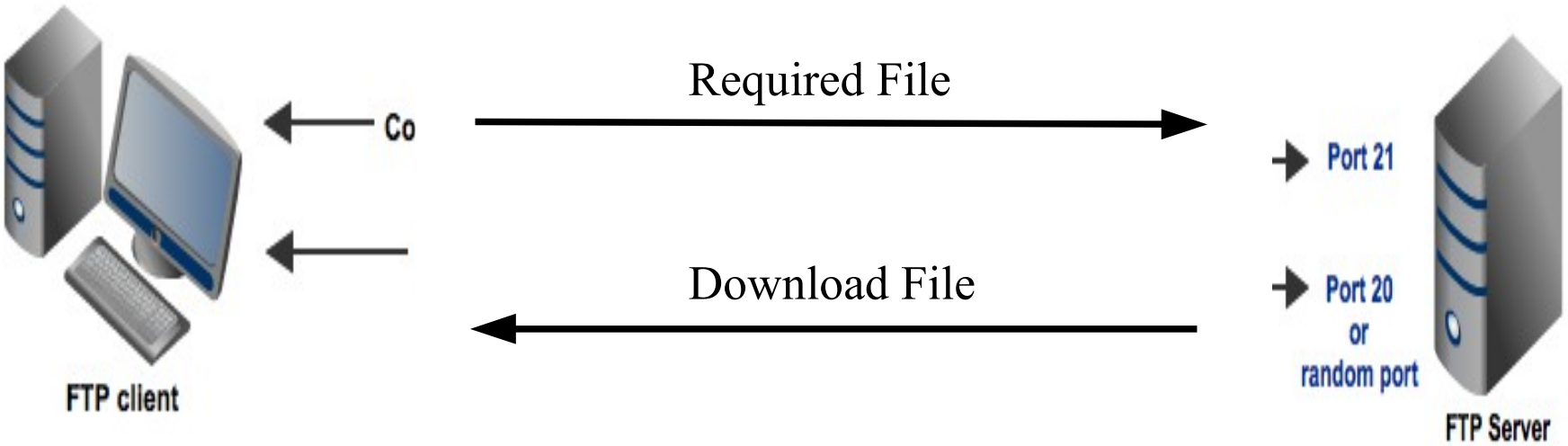
Get and parse JSON file

```
JSONObject obj = (JSONObject) new JSONParser().parse(sb.toString());
JSONArray results = (JSONArray) obj.get("items");
//Leggo i primi K link
if(numeroRisultati<=results.size()){
    for(int i=0;i<numeroRisultati;i++){
        JSONObject post=(JSONObject) results.get(i);
        String linkImage= (String) ((JSONObject) post.get("media")).get("m");
        System.out.println("get images : "+linkImage);
        pool.execute(new GetImage(linkImage));
    }
    pool.shutdown();
}
```

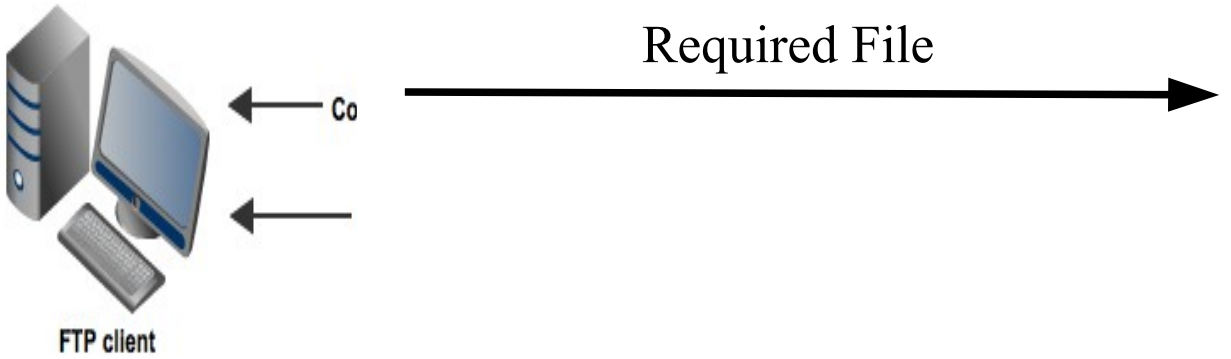
Download images concurrently

Mini FTP

- Overview FTP

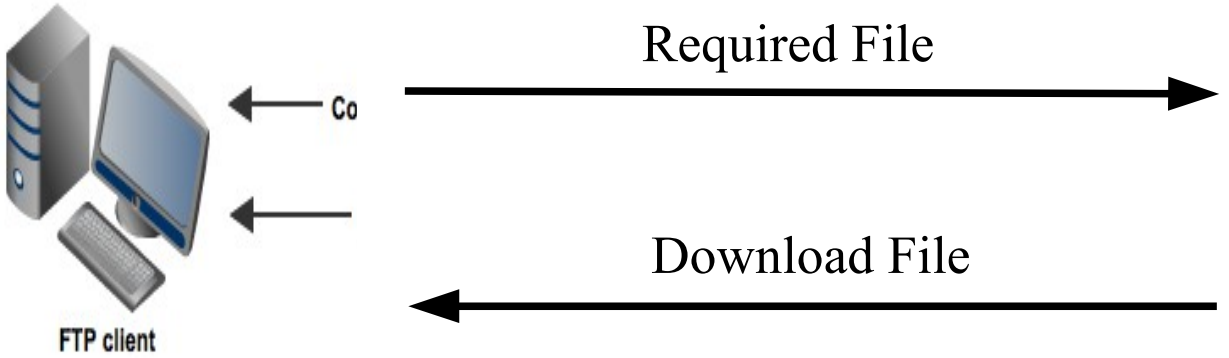


The Mini FTP Client (1)




```
public class FTPClient {

    public static void main( String[] args ) {

        String s, filename, hostname = "localhost";
        int port = FTPServer.PORT;
        filename = "Testo.txt";
        try {
            Socket sck = new Socket ( hostname, port );
            //Leggo file name
            BufferedWriter bw = new BufferedWriter(
                    new OutputStreamWriter( sck.getOutputStream() ));
            BufferedReader br = new BufferedReader(
                    new InputStreamReader( sck.getInputStream() ));

            bw.write(filename);
            bw.newLine();
            bw.flush();
```

The Mini FTP Client (2)

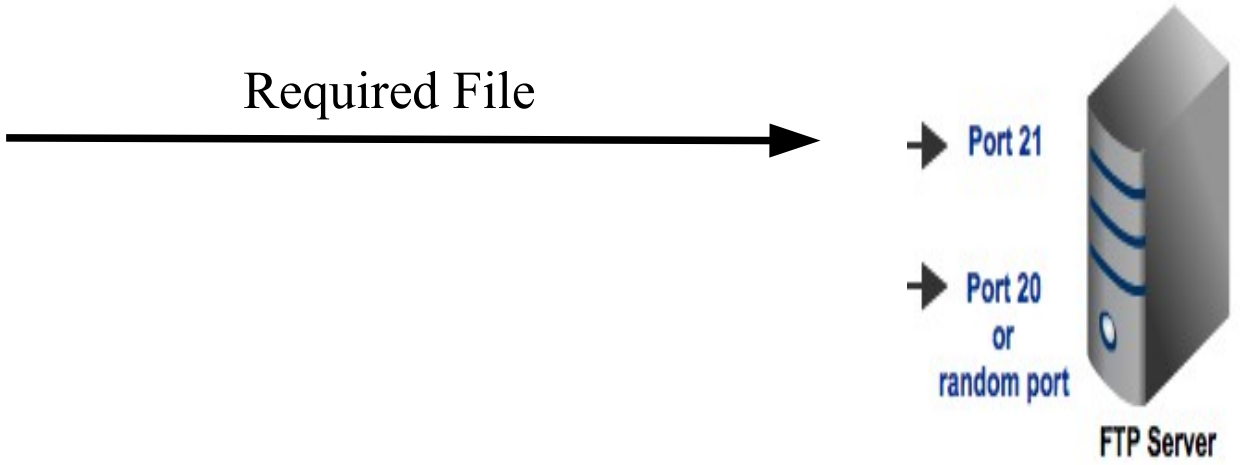



```
PrintWriter pw=new PrintWriter(filename+"FTP");
while( (s=br.readLine()) != null ) {
    System.out.println(s);
    pw.println(s);
}
sck.close();
pw.close();
```

The Mini FTP Server (1)

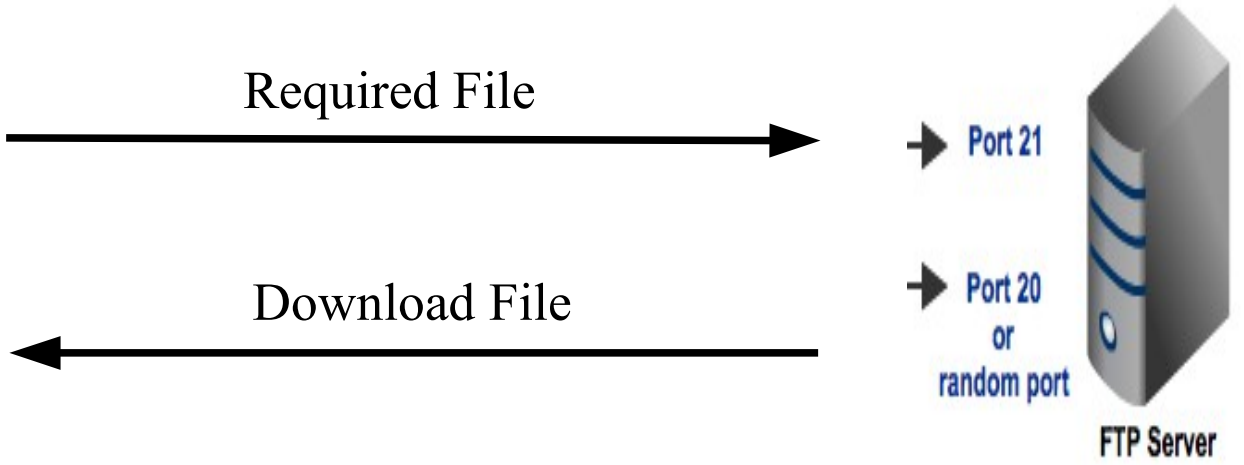

```
public class FTPServer implements Runnable{

    public static int PORT=4444;
    public static int FILE_NOT_FOUND=0;

    //--------------------
    //  Runnable field
    //--------------------
    private Socket sck;

    FTPServer(Socket sck){
        this.sck=sck;
    }

    public void run() {
```

The Mini FTP Server (2)



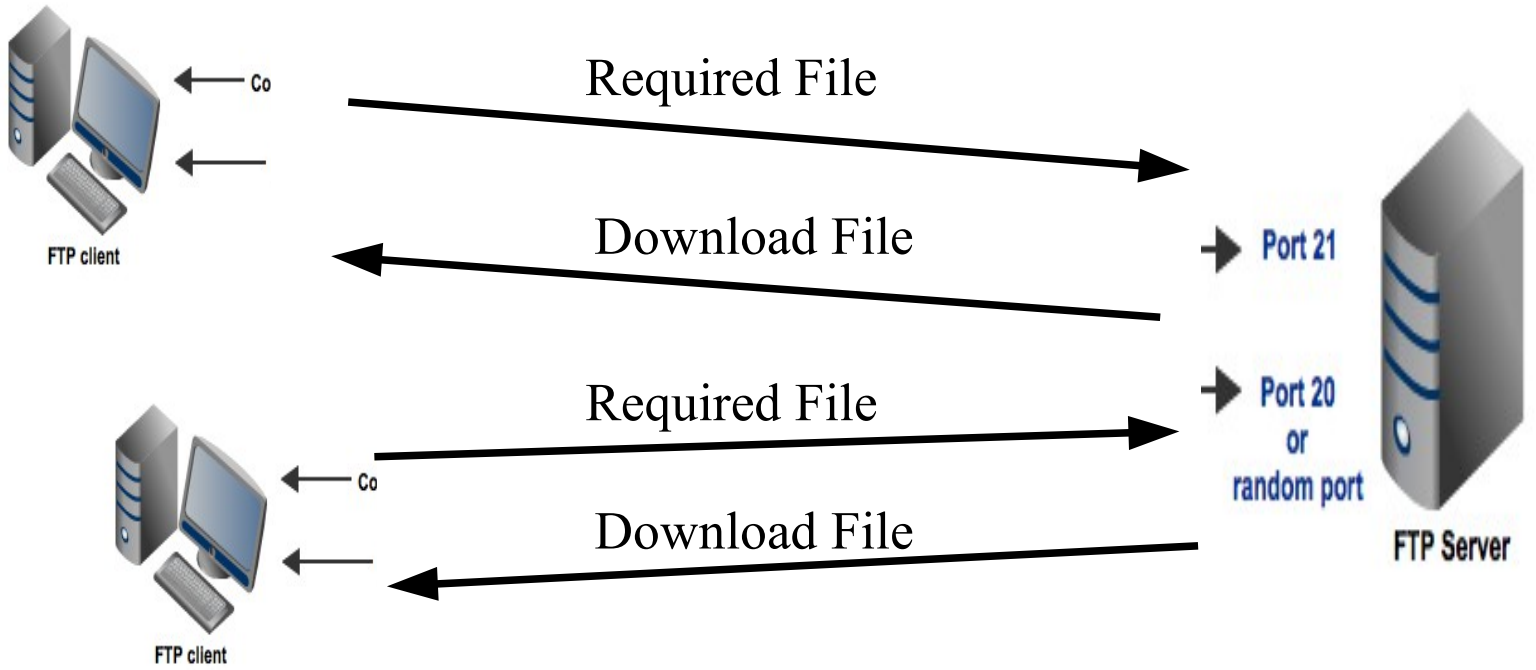

```
public void run() {
    try {
        BufferedReader rd = new BufferedReader(
                new InputStreamReader(sck.getInputStream()));
        BufferedWriter wr = new BufferedWriter(
                new OutputStreamWriter(sck.getOutputStream()));

        //Nome File
        String line = rd.readLine();

        try {
            BufferedReader br=new BufferedReader(new FileReader(line));
            //Send File
```

The Mini FTP Server (3)




```
        String line = rd.readLine();

        try {
            BufferedReader br=new BufferedReader(new FileReader(line));
            //Send File
            while ( (line=br.readLine()) != null ) {
                wr.write(line);
                wr.newLine();
                wr.flush();
                line = br.readLine();
            }
            br.close();
        }
        catch ( FileNotFoundException e ) { e.printStackTrace();  }
        rd.close(); wr.close();
    }
    catch ( Exception e ) { e.printStackTrace(); }
}
```

The Mini FTP Server Main




```
public static void main( String[] args ) throws Exception {
    ServerSocket sck = new ServerSocket(PORT);
    ThreadPoolExecutor pool=(ThreadPoolExecutor) Executors.newCachedThreadPool();
    System.out.println("FTP Server started...");
    boolean stop=false;
    while(!stop) {
        Socket socket = sck.accept();
        pool.execute(new FTPServer(socket));
    }
    sck.close();
}
```

Assignments corrected

- Alarm Clock
- Pi Greco
- Ufficio Postale
- Gestione Auto

- WebLookUp
- Calcolatrice TCP
- Mini FTP
- Query Flickr

……

➔ Topics

Java Concurrent

Java Basic I/O

Java Networking

More Info

- Java Tutorial: https://docs.oracle.com/javase/tutorial/
- JavaDoc:

  https://docs.oracle.com