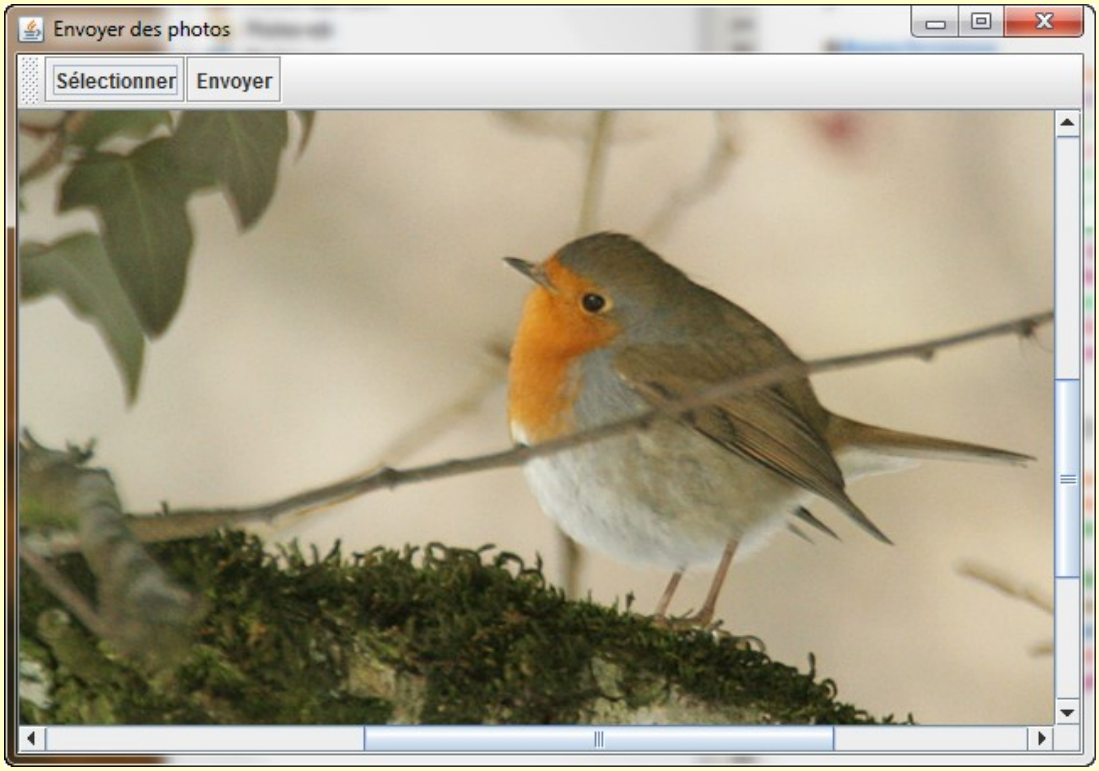
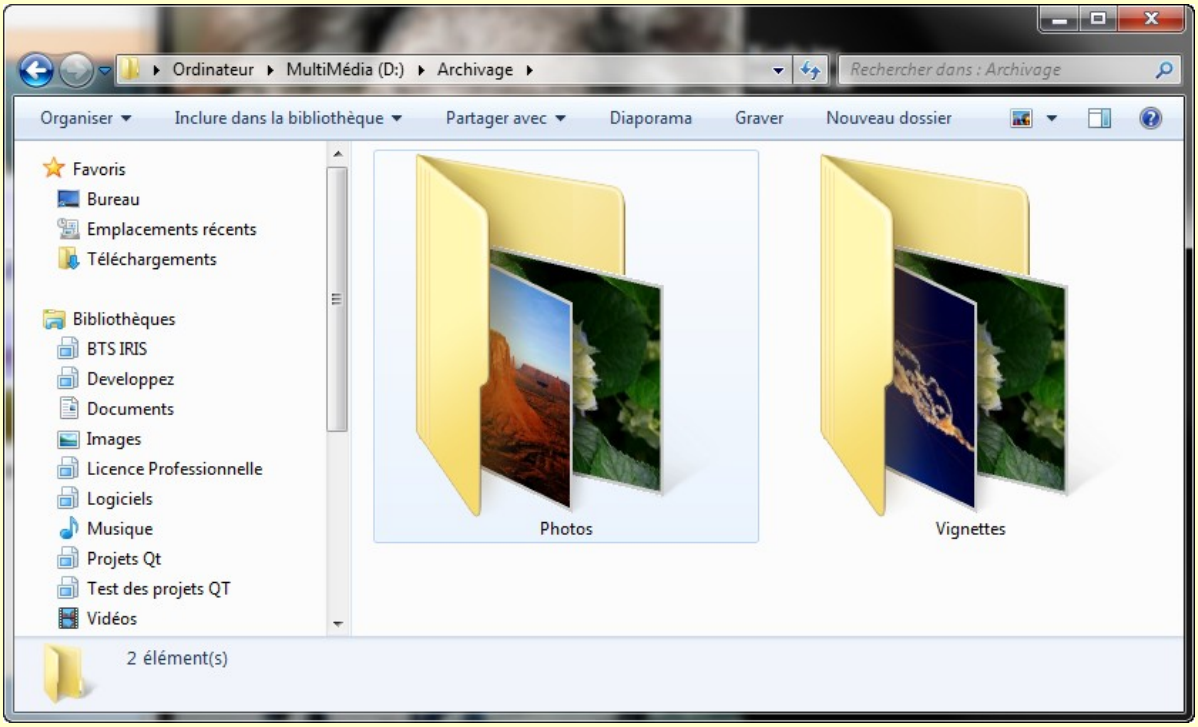
## JAVA EE 6

Au travers de ce TP, nous allons mettre en œuvre un projet d'entreprise qui archive des photos sur le disque dur du serveur, afin de pouvoir les récupérer ultérieurement.

- *Pour cela, nous devons créer deux applications clientes fenêtrées, d'une part en réseau local, et également par service Web qui permettent d'envoyer les photos sélectionnées.*

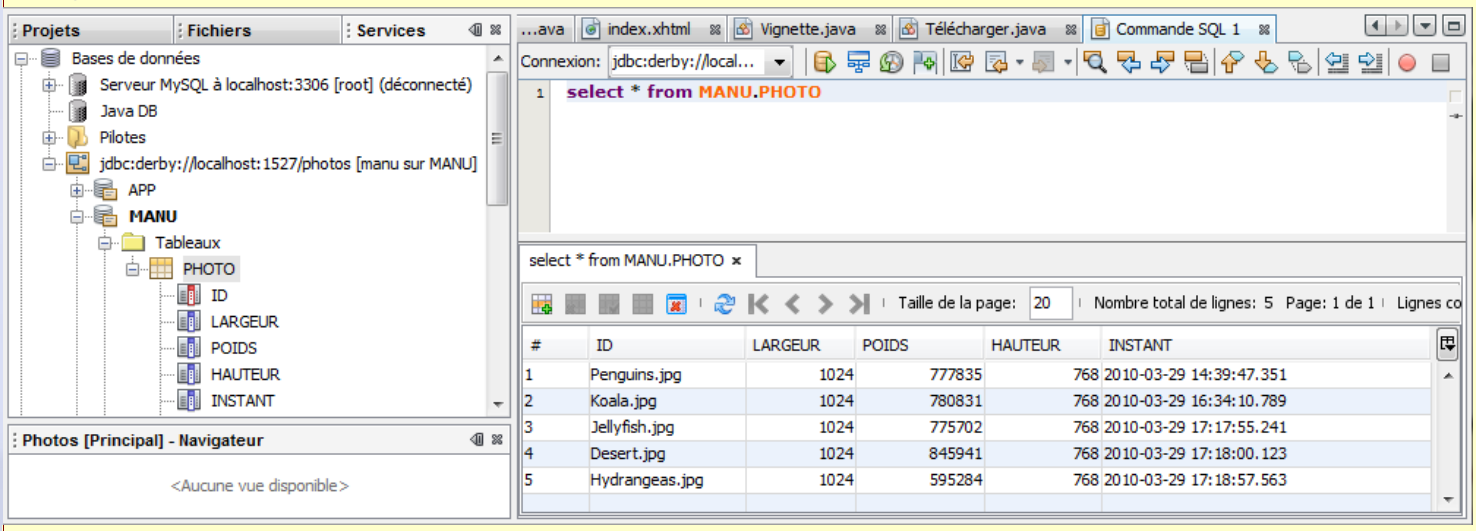



- *Côté serveur, nous implémentons un bean session **Archiver** qui permet d'archiver les photos reçues, d'une part en taille réelle, mais également sous forme de vignettes afin que la consultation puisse se faire ultérieurement très rapidement.*

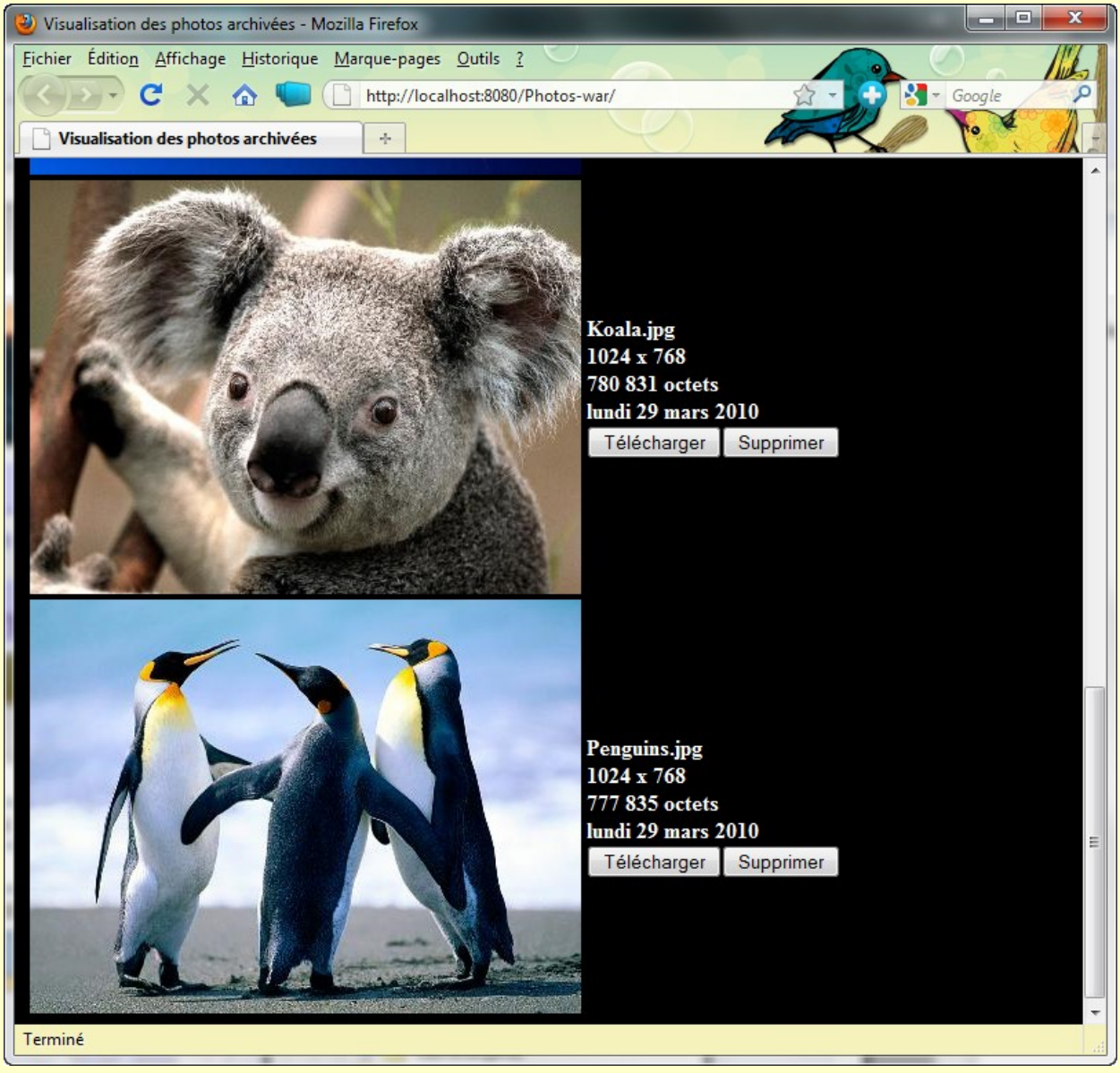



- *Nous devons également créer un autre bean session **ArchiverWS** qui assure le web service requis en utilisant les compétences du bean session **Archiver**.*

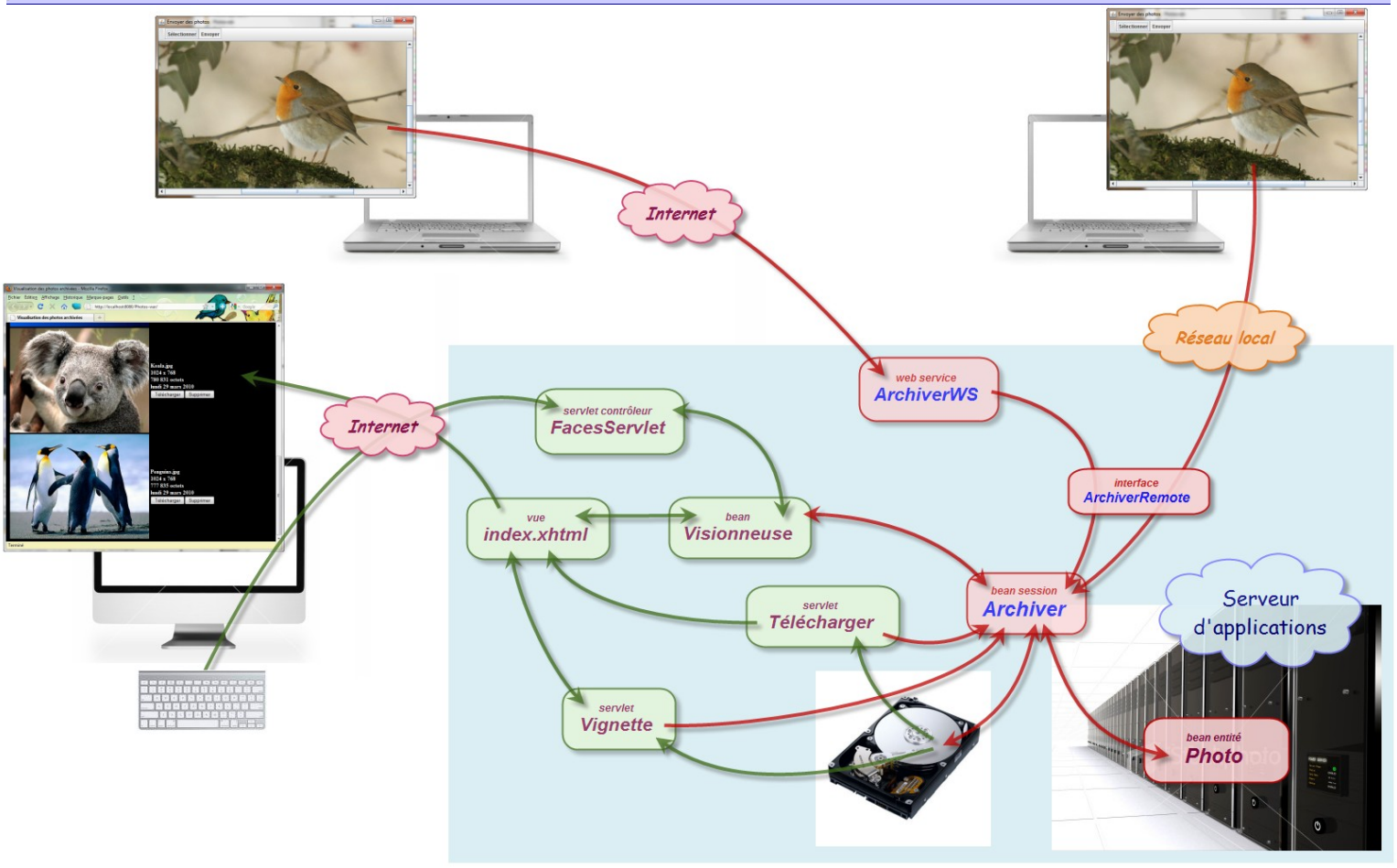
- *Toujours côté serveur, nous désirons enregistrer quelques caractéristiques de la photo archivée au moyen d'un bean entité **Photo** adapté.*




- *Enfin, nous avons besoin de mettre en place un autre client au travers d'une application Web qui permet à la fois d'afficher les photos présentes dans le serveur sous forme de vignette, avec les différentes caractéristiques, et de télécharger la version originale choisie.*



## ARCHITECTURE GLOBALE DU SYSTÈME




## WEB SERVICE

E:\BTS IRIS\TP Java\Photos\Photos-ejb\src\java\ws\ArchiverWS.java

```
1 package ws;
2
3 import java.io.IOException;
4 import javax.ejb.EJB;
5 import javax.jws.*;
6 import javax.ejb.Stateless;
7 import session.ArchiverRemote;
8
9 @WebService
10 @Stateless
11 public class ArchiverWS {
12    @EJB
13    private ArchiverRemote archivage;
14
15    public void envoyer(String nom, Byte[] contenu) throws IOException {
16       byte[] octets = new byte[contenu.length];
17       for (int i=0; i<contenu.length; i++) octets[i] = contenu[i];
18       archivage.stocker(nom, octets);
19    }
20 }
21
```

## INTERFACE

E:\BTS IRIS\TP Java\Photos\Photos-ejb\src\java\session\ArchiverRemote.java

```
1 package session;
2
3 import java.io.IOException;
4 import javax.ejb.Remote;
5
6 @Remote
7 public interface ArchiverRemote {
8    void stocker(String nom, byte[] octets) throws IOException;
9 }
10
11
```

## BEANS ENTITÉ

E:\BTS IRIS\TP Java\Photos\Photos-ejb\src\java\entité\Photo.java

```
1 package entité;
2
3 import java.awt.image.BufferedImage;
4 import java.io.Serializable;
5 import java.util.Date;
6 import javax.persistence.*;
7
8 @Entity
9 public class Photo implements Serializable {
10    @Id
11    private String id;
12    @Temporal(TemporalType.TIMESTAMP)
13    private Date instant;
14    private int largeur;
15    private int hauteur;
16    private long poids;
17
18    public String getId() { return id; }
19    public int getHauteur() { return hauteur;  }
20    public Date getInstant() { return instant;  }
21    public int getLargeur() { return largeur;  }
22    public long getPoids() { return poids;  }
23
24    public Photo() { }
25
26    public Photo(String nom, long poids) {
27       id = nom;
28       instant = new Date();
29       this.poids = poids;
30    }
31
32    public void setDimensions(BufferedImage image) {
33       largeur = image.getWidth();
34       hauteur = image.getHeight();
35    }
36 }
37
38
```

## BEANS SESSION

E:\BTS IRIS\TP Java\Photos\Photos-ejb\src\java\session\Archiver.java

```
1 package session;
2
3 import entité.Photo;
4 import java.awt.geom.AffineTransform;
5 import java.awt.image.*;
6 import java.io.*;
7 import javax.ejb.*;
8 import javax.imageio.ImageIO;
9 import javax.persistence.*;
10
11 @Stateless
12 @LocalBean
13 public class Archiver implements ArchiverRemote {
14     private final String répertoire = "D:/Archivage/";
15     private final String repPhotos = répertoire + "Photos/";
16     private final String repVignettes = répertoire + "Vignettes/";
17     private final double largeur = 400.0;
18     @PersistenceContext
19     EntityManager persistance;
20
21     public void stocker(String nom, byte[] octets) throws IOException  {
22         File fichier = new File(repPhotos+nom);
23         if (fichier.exists()) return;
24         FileOutputStream photo = new FileOutputStream(fichier);
25         photo.write(octets);
26         photo.close();
27         fabriquerVignette(nom);
28     }
29
30     @Asynchronous
31     private void fabriquerVignette(String nom) throws IOException {
32         File fichier = new File(repPhotos+nom);
33         BufferedImage photo = ImageIO.read(fichier);
34         double ratio = photo.getWidth() / (double) largeur;
35         BufferedImage vignette = new BufferedImage((int)(photo.getWidth()/ratio), (int)(photo.getHeight()/ratio), photo.getType());
36         AffineTransform échelle = AffineTransform.getScaleInstance(1/ratio, 1/ratio);
37         AffineTransformOp retailler = new AffineTransformOp(échelle, AffineTransformOp.TYPE_BICUBIC);
38         retailler.filter(photo, vignette);
39         ImageIO.write(vignette, "JPEG", new FileOutputStream(repVignettes+nom));
40         Photo caractéristiques = new Photo(nom, fichier.length());
41         caractéristiques.setDimensions(photo);
42         persistance.persist(caractéristiques);
43     }
44
45     public String[] getListe() { return new File(repPhotos).list(); }
46
47     public String getRepVignettes() { return repVignettes; }
48     public String getRepPhotos() { return repPhotos; }
49     public Photo getPhoto(String nom) {  return persistance.find(Photo.class, nom);  }
50
51     public void supprimer(String nom) {
52         new File(repPhotos+nom).delete();
53         new File(repVignettes+nom).delete();
54         Photo photo = getPhoto(nom);
55         persistance.remove(photo);
56     }
57 }
```

## x APPLICATION CLIENTE EN RÉSEAU LOCAL

E:\BTS IRIS\TP Java\Photos\Photos-app-client\src\java\photos\Client.java

```
1 package photos;
2
3 import java.awt.BorderLayout;
4 import java.awt.event.ActionEvent;
5 import java.io.*;
6 import javax.ejb.EJB;
7 import javax.swing.*;
8 import session.ArchiverRemote;
9
10 public class Client extends JFrame {
11    private JToolBar outils = new JToolBar();
12    private JLabel photo = new JLabel();
13    private JFileChooser sélecteur = new JFileChooser();
14    private File fichier;
15    @EJB private static ArchiverRemote archivage;
16
17    public Client() {
18       super("Envoyer des photos");
19       add(outils, BorderLayout.NORTH);
20       add(new JScrollPane(photo));
21       outils.add(new AbstractAction("Sélectionner") {
22          public void actionPerformed(ActionEvent e) {
23             if (sélecteur.showOpenDialog(Client.this)==JFileChooser.APPROVE_OPTION) {
24                fichier = sélecteur.getSelectedFile();
25                photo.setIcon(new ImageIcon(fichier.getPath()));
26             }
27          }
28       });
29       outils.add(new AbstractAction("Envoyer") {
30          public void actionPerformed(ActionEvent e) {
31             if (fichier!=null)
32                try {
33                   byte[] octets = new byte[(int) fichier.length()];
34                   FileInputStream lecture = new FileInputStream(fichier);
35                   lecture.read(octets);
36                   lecture.close();
37                   archivage.stocker(fichier.getName(), octets);
38                }
39                catch (Exception ex) {
40                   setTitle("Impossible d'envoyer le fichier");
41                }
42             }
43
44       });
45       setSize(500, 400);
46       setLocationByPlatform(true);
47       setDefaultCloseOperation(EXIT_ON_CLOSE);
48       setVisible(true);
49    }
50
51    public static void main(String[] args) { new Client(); }
52 }
53
```

## APPLICATION CLIENTE EN CONNEXION AVEC LE WEB SERVICE

E:\BTS IRIS\TP Java\PhotosWS\src\photosws\Client.java

```
1 package photosws;
2
3 import java.awt.BorderLayout;
4 import java.awt.event.ActionEvent;
5 import java.io.*;
6 import java.util.*;
7 import javax.swing.*;
8 import ws.*;
9
10 public class Client extends JFrame {
11    private JToolBar outils = new JToolBar();
12    private JLabel photo = new JLabel();
13    private JFileChooser sélecteur = new JFileChooser();
14    private File fichier;
15    private static ArchiverWS archivage;
16
17    public Client() {
18       super("Envoyer des photos");
19       add(outils, BorderLayout.NORTH);
20       add(new JScrollPane(photo));
21       outils.add(new AbstractAction("Sélectionner") {
22          public void actionPerformed(ActionEvent e) {
23             if (sélecteur.showOpenDialog(Client.this)==JFileChooser.APPROVE_OPTION) {
24                fichier = sélecteur.getSelectedFile();
25                photo.setIcon(new ImageIcon(fichier.getPath()));
26             }
27          }
28       });
29       outils.add(new AbstractAction("Envoyer") {
30          public void actionPerformed(ActionEvent e) {
31             if (fichier!=null)
32                try {
33                   byte[] octets = new byte[(int) fichier.length()];
34                   FileInputStream lecture = new FileInputStream(fichier);
35                   lecture.read(octets);
36                   lecture.close();
37                   List<Byte> contenu = new ArrayList<Byte>();
38                   for (byte octet : octets) contenu.add(octet);
39                   archivage.envoyer(fichier.getName(), contenu);
40                }
41                catch (Exception ex) {
42                   setTitle("Impossible d'envoyer le fichier");
43                }
44             }
45
46       });
47       setSize(500, 400);
48       setLocationByPlatform(true);
49       setDefaultCloseOperation(EXIT_ON_CLOSE);
50       setVisible(true);
51    }
52
53    public static void main(String[] args) {
54       archivage = new ArchiverWSService().getArchiverWSPort();
55       new Client();
56    }
57 }
58
```

## APPLICATION CLIENT WEB

E:\BTS IRIS\TP Java\Photos\Photos-war\src\java\bean\Visionneuse.java

```
1 package bean;
2
3 import entité.Photo;
4 import java.io.*;
5 import javax.ejb.EJB;
6 import javax.faces.bean.*;
7 import javax.faces.component.UIData;
8 import javax.faces.context.*;
9 import session.Archiver;
10
11 @ManagedBean(name="visu")
12 @SessionScoped
13 public class Visionneuse {
14    @EJB private Archiver archive;
15    private UIData table;
16    private String[] liste;
17
18    public UIData getTable() { return table; }
19    public void setTable(UIData table) { this.table = table;  }
20
21    public String[] getListe() { return liste = archive.getListe();  }
22    public Photo getPhoto() { return archive.getPhoto(liste[table.getRowIndex()]); }
23
24    public void supprimer() { archive.supprimer(liste[table.getRowIndex()]); }
25
26    public void téléchargement() {
27       ExternalContext contexte = FacesContext.getCurrentInstance().getExternalContext();
28       String nom = liste[table.getRowIndex()];
29       try {
30          contexte.redirect("/Photos-war/Telecharger/"+nom+"?nom="+nom);
31       }
32       catch (IOException ex) {  }
33    }
34 }
```

E:\BTS IRIS\TP Java\Photos\Photos-war\src\java\servlet\Vignette.java

```
1 package servlet;
2
3 import java.io.*;
4 import javax.ejb.EJB;
5 import javax.servlet.ServletException;
6 import javax.servlet.annotation.WebServlet;
7 import javax.servlet.http.*;
8 import session.Archiver;
9
10 @WebServlet(name="Vignette", urlPatterns={"/Vignette"})
11 public class Vignette extends HttpServlet {
12    @EJB Archiver archive;
13
14    @Override
15    protected void doGet(HttpServletRequest request, HttpServletResponse response)
16    throws ServletException, IOException {
17       response.setContentType("image/jpeg");
18       String nom = request.getParameter("nom");
19       File fichier = new File(archive.getRepVignettes()+nom);
20       byte[] octets = new byte[(int)fichier.length()];
21       FileInputStream vignette = new FileInputStream(fichier);
22       vignette.read(octets);
23       vignette.close();
24       response.getOutputStream().write(octets);
25    }
26 }
```

E:\BTS IRIS\TP Java\Photos\Photos-war\src\java\servlet\Télécharger.java

```java
package servlet;

import java.io.*;
import javax.ejb.EJB;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.*;
import session.Archiver;

@WebServlet(name="Télécharger", urlPatterns={"/Telecharger/*"})
public class Télécharger extends HttpServlet {
    @EJB Archiver archive;

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("application/octet-stream");
        String nom = request.getParameter("nom");
        File fichier = new File(archive.getRepPhotos()+nom);
        byte[] octets = new byte[(int)fichier.length()];
        FileInputStream photo = new FileInputStream(fichier);
        photo.read(octets);
        photo.close();
        response.getOutputStream().write(octets);
    }
}
```

E:\BTS IRIS\TP Java\Photos\Photos-war\web\index.xhtml

```xml
<?xml version='1.0' encoding='UTF-8' ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://java.sun.com/jsf/html"
      xmlns:c="http://java.sun.com/jsf/core">
    <h:head>
        <title><h:outputText value="Visualisation des photos archivées" /></title>
    </h:head>

    <style type="text/css">
        body {
            background-color: black;
            color: white;
            font-weight: bold;
        }
    </style>

    <h:body>
        <h2 align="center"><h:outputText value="Ensemble des photos archivées"/></h2>
        <hr />
        <h:form>
            <h:dataTable value="#{visu.liste}" var="nom" binding="#{visu.table}">
                <h:column><h:graphicImage value="Vignette?nom=#{nom}" /></h:column>
                <h:column>
                    <h:outputText value="#{nom}" /><br />
                    <h:outputText value="#{visu.photo.largeur} x #{visu.photo.hauteur}" /><br />
                    <h:outputText value="#{visu.photo.poids}">
                        <c:convertNumber pattern="#,##0 octets" />
                    </h:outputText><br />
                    <h:outputText value="#{visu.photo.instant}">
                        <c:convertDateTime dateStyle="full" />
                    </h:outputText><br />
                    <h:commandButton value="Télécharger" action="#{visu.téléchargement}"/>
                    <h:commandButton value="Supprimer" action="#{visu.supprimer}"/>
                </h:column>
            </h:dataTable>
        </h:form>
    </h:body>
</html>
```