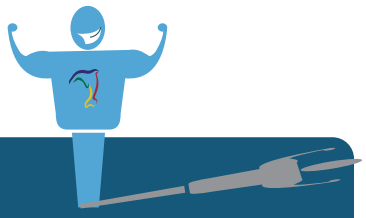
autentia

Avenida de Castilla,1 - Edificio Best Point - Oficina 21B
28830 San Fernando de Henares (Madrid)
tel./fax: +34 91 675 33 06
info@autentia.com - **www.autentia.com**

¿**Qué ofrece** Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
Ese apoyo que siempre quiso tener...

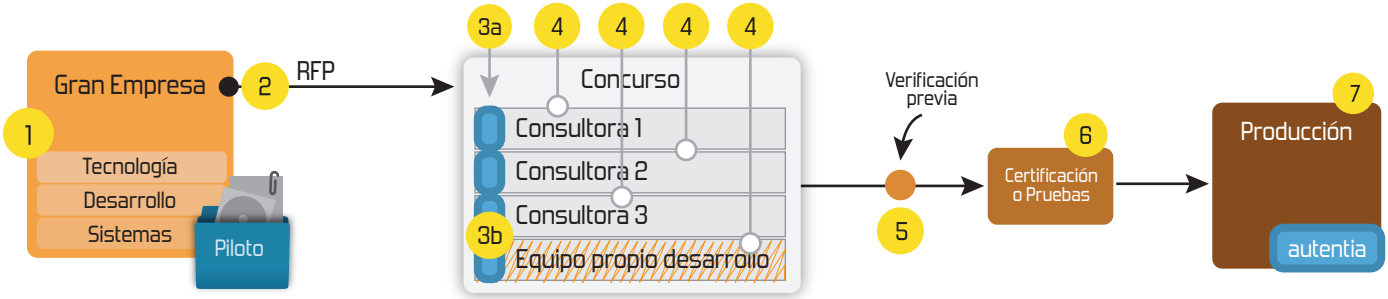
1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces, HTML5, CSS3, JavaScript-jQuery

Gestor portales (Liferay)
Gestor de contenidos (Alfresco)
Aplicaciones híbridas

Control de autenticación y acceso (Spring Security)
UDDI
Web Services
Rest Services
Social SSO
SSO (Cas)

JPA-Hibernate, MyBatis
Motor de búsqueda empresarial (Solr)
ETL (Talend)

Dirección de Proyectos Informáticos.
Metodologías ágiles
Patrones de diseño
TDD

Tareas programadas (Quartz)
Gestor documental (Alfresco)
Inversión de control (Spring)

BPM (jBPM o Bonita)
Generación de informes (JasperReport)
ESB (Open ESB)

Compartimos nuestro conocimiento en:
www.adictosaltrabajo.com

Para más información visítenos en:
www.autentia.com

Descargar este documento en formato PDF fileupload.pdf

FileUpload: Subida de Ficheros al Servidor

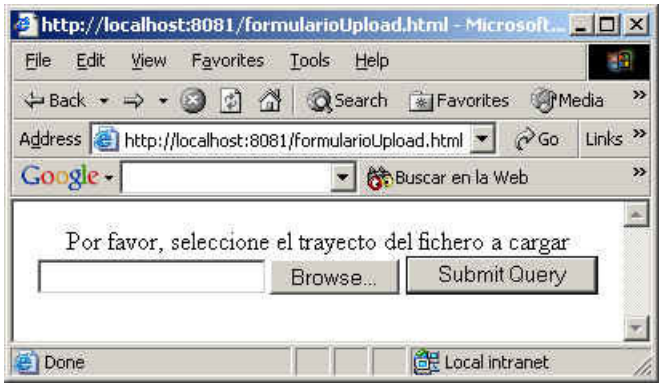
Cuando realizamos una aplicación Java, es posible que tengamos la necesidad de crear un sistema de administración, normalmente en la Intranet, para simplificar las labores de mantenimiento.

Esta administración, posiblemente estará formada por unos cuantos formularios, normalmente interactuará con una base de datos.... pero a veces, también tenemos la necesidad de subir ficheros al servidor.

Para subir un fichero al servidor, debemos crear un formulario un poco especial... via POST e indicando que es una subida multi-parte.

```
<HTML>
<HEAD>
<TITLE></TITLE>
</HEAD>
<BODY>
<center>
<form method="POST" enctype='multipart/form-data' action="/servlet/uploadFichero">
Por favor, seleccione el trayecto del fichero a cargar
<br><input type="file" name="fichero">
<input type="submit">
</form>
</center>
</BODY>
</HTML>
```

En el formulario anterior, podemos ver que vamos a subir la respuesta a un servlet llamado uploadFichero... este es el aspecto del formulario.



En otro tutorial, vimos como crear un servidor Web.... si atacasemos a nuestro servidor (cambiando el destino de nuestro formulario), obtendriamos una respuesta como ésta.... con ello nos hacemos una idea de lo que el servidor se encuentra:

```
Thread[Thread-1,4,main] - --POST / HTTP/1.1-
Thread[Thread-1,4,main] - --Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, application/vnd.ms-powerpoint,
application/vnd.ms-excel, application/msword, application/x-shockwave-flash, */*-
Thread[Thread-1,4,main] - --Referer: http://localhost:8081/formularioUpload.html-
Thread[Thread-1,4,main] - --Accept-Language: es-
Thread[Thread-1,4,main] - --Content-Type: multipart/form-data; boundary=----------------------------7d31df1ed049a-
Thread[Thread-1,4,main] - --Accept-Encoding: gzip, deflate-
Thread[Thread-1,4,main] - --User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0; .NET CLR 1.1.4322)-
Thread[Thread-1,4,main] - --Host: localhost:90-
Thread[Thread-1,4,main] - --Content-Length: 325-
Thread[Thread-1,4,main] - --Connection: Keep-Alive-
Thread[Thread-1,4,main] - --Cache-Control: no-cache-
Thread[Thread-1,4,main] - ---
Thread[Thread-1,4,main] - ------------------------------7d31df1ed049a-
Thread[Thread-1,4,main] - --Content-Disposition: form-data; name="fichero"; filename="C:\Leeme.txt"-
Thread[Thread-1,4,main] - --Content-Type: text/plain-
Thread[Thread-1,4,main] - ---
Thread[Thread-1,4,main] - --Este es un ejemplo de un fichero que vamos a tratar de hacer el upload.-
Thread[Thread-1,4,main] - --Para ello, trataremos de simplificar la labor al máximo-
Thread[Thread-1,4,main] - ---
Thread[Thread-1,4,main] - ---
Thread[Thread-1,4,main] - ------------------------------7d31df1ed049a---
```

Es decir ..... vemos que es servidor es capaz de recibir una respuesta y nosotros en nuestro servlet debe hacer lo mismo.

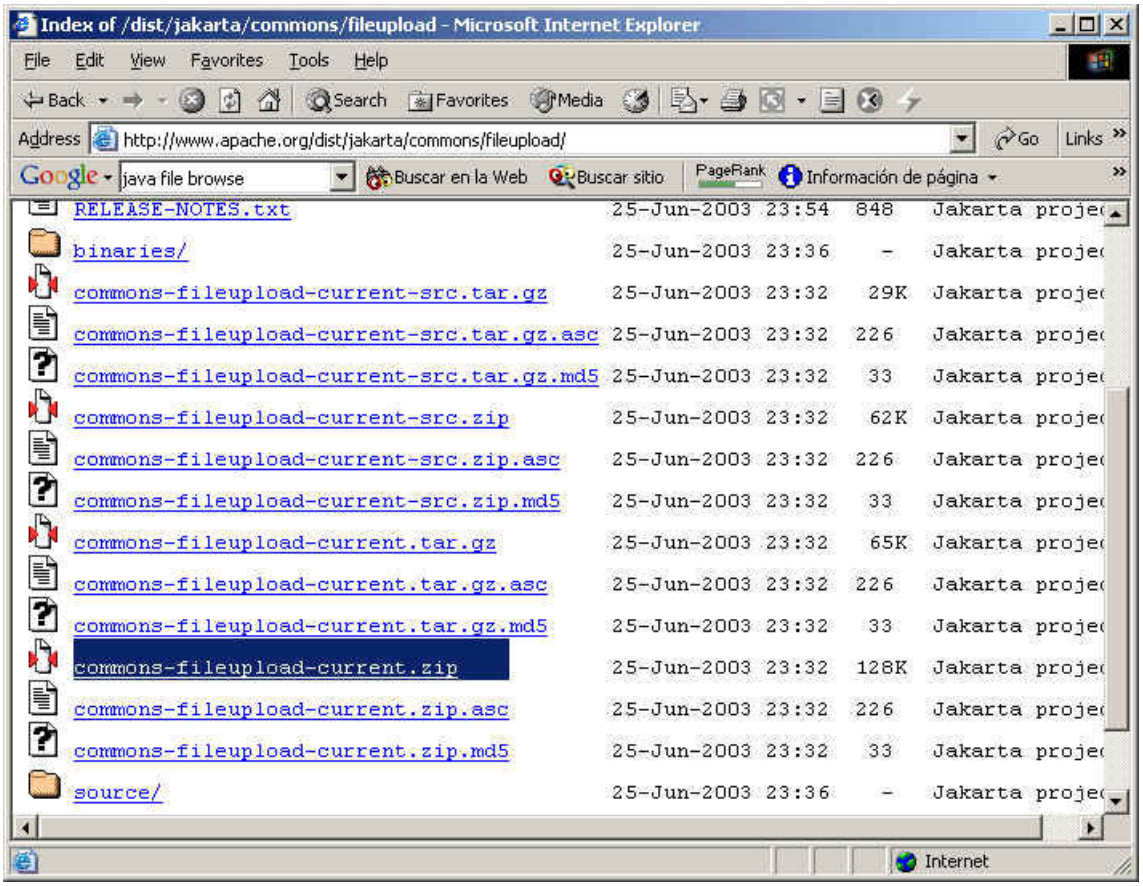
Ahora, tenemos que crear un servlet que sea capaz de procesar la respuesta....

La labor ... no parece demasiado sencilla .... aunque siempre hay alguien que nos facilitará la tarea ....

En este caso .... hay un subproyecto en apache http://jakarta.apache.org/commons/fileupload/

que nos proporciona las clases para que nuestro trabajo sea mínimo:

Nos descargamos el jar (no olvidar incluirlo en el classpath). Si os falla este proyecto.... seguro que es por ésto.



Solamente tenemos que escribir un código mínimo

```
/*
 * uploadFichero.java
 *
 * Created on 4 de agosto de 2003, 22:26
 */

import java.io.*;
import java.net.*;

import javax.servlet.*;
import javax.servlet.http.*;

import org.apache.commons.fileupload.*;
import java.util.*;
/**
 *
 * @author  Roberto Canales
 * @version
 */
public class uploadFichero extends HttpServlet {

    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet</title>");
        out.println("</head>");
        out.println("<body>");

        System.out.println("Comenzamos procesamiento ficheros");

        procesaFicheros(request,out);

        out.println("</body>");
        out.println("</html>");

        out.close();
    }


    void depura(String cadena)
    {
        System.out.println("El error es " + cadena);
    }

    public boolean procesaFicheros(HttpServletRequest req, PrintWriter out ) {
        try {
```

```
        // construimos el objeto que es capaz de parsear la perición
        DiskFileUpload fu = new DiskFileUpload();

        // maximo numero de bytes
        fu.setSizeMax(1024*512); // 512 K

        depura("Ponemos el tamaño máximo");
        // tamaño por encima del cual los ficheros son escritos directamente en disco
        fu.setSizeThreshold(4096);

        // directorio en el que se escribirán los ficheros con tamaño superior al soportado en memoria
        fu.setRepositoryPath("/tmp");

        // ordenamos procesar los ficheros
        List fileItems = fu.parseRequest(req);

        if(fileItems == null)
        {
           depura("La lista es nula");
           return false;
        }

        out.print("<br>El número de ficheros subidos es: " +  fileItems.size());

        // Iteramos por cada fichero

              Iterator i = fileItems.iterator();
        FileItem actual = null;
        depura("estamos en la iteración");

        while (i.hasNext())
        {
           actual = (FileItem)i.next();
           String fileName = actual.getName();
           out.println("<br>Nos han subido el fichero" + fileName);

           // construimos un objeto file para recuperar el trayecto completo
           File fichero = new File(fileName);
           depura("El nombre del fichero es " + fichero.getName());

           // nos quedamos solo con el nombre y descartamos el path
           fichero = new  File("c:\\ficherossubidos\\" + fichero.getName());

           // escribimos el fichero colgando del nuevo path
           actual.write(fichero);
        }

     }
     catch(Exception e) {
        depura("Error de Aplicación " + e.getMessage());
        return false;
     }

     return true;
  }

  /** Handles the HTTP <code>POST</code> method.
   * @param request servlet request
   * @param response servlet response
   */
  protected void doPost(HttpServletRequest request, HttpServletResponse response)
  throws ServletException, IOException {
     processRequest(request, response);
  }
}
```

El resultado es impresionante .. en el directorio que hemos seleccionado.... tenemos el fichero que hemos solicitado ..... y nos a costado apenas unas lineas...

En proximos tutorial os hablaremos de otras de las APIS que están disponibles en apache ....

Sobre el Autor ..

Nuevo servicio de notificaciones

Si deseas que te enviemos un correo electrónico cuando introduzcamos nuevos tutoriales, inserta tu dirección de correo en el siguiente formulario.

Subscribirse a Novedades	
e-mail	
	Enviar

Otros Tutoriales Recomendados (También ver todos)

Nombre Corto	Descripción
Comunicación entre Applets y Servlets	Os mostramos como comunicar un applet y un servlet a través de GET y POST, serializando objetos y teniendo en cuenta proxys y autentificación
Gestión de contenidos y errores comunes	Os explicamos en que consiste la gestión de contenidos y cuales son los errores cometidos por multitud de empresas a la hora de abordar su implantación
Transformación de XML y XSL en JSPs	Os mostramos como poder utilizar XML y XSL en JSPS, combinado con el Patrón MVC
Filtros de Servlets en Tomcat	En este tutorial os enseñamos la técnica (poco conocida) del encadenamiento de filtros en la activación de servlets, dentro del entorno Tomcat
Pool de Conexiones y Tomcat5	Os mostramos como instalar Tomcat5 en vuestro PC y como ejemplo de uso, configuramos un Pool de Conexiones y lo usamos contra MySQL
TagLibs y JSPs	Os mostramos como crear librerias de etiquetas para vuestros JSP y así simplificar su construcción.
Generar imagenes desde Servlets	Os mostramos como generar ficheros GIF desde un servlet java. Util para generear gráficas dinámicas, contadores, etc
JSP 2.0, JSTL y Lenguaje de expresiones	Os mostramos las novedades de JSP 2.0: Nuevas librerías estandar de etiquetas y el lenguaje de expresiones con ejemplos de acceso a base de datos, XML y XSL en JSP
Comunicación entre TAGs, Beans y JSPs	Os mostramos las posibilidades de comunicación entre JSPs, Bean y etiquetas de usuario.

www.AdictosAlTrabajo.com Opimizado 800X600