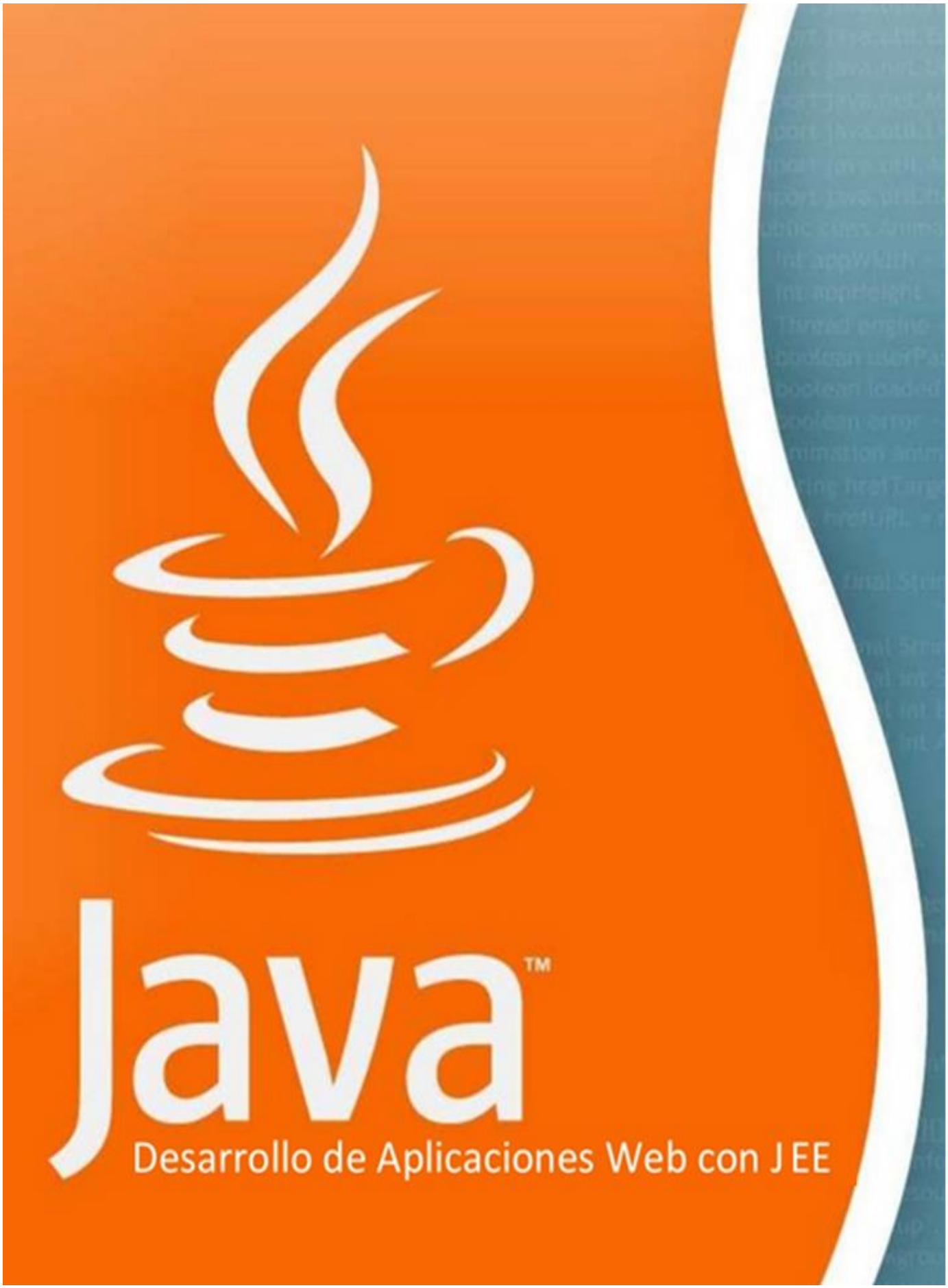
Java™
Desarrollo de Aplicaciones Web con J EE

PARTE VIII

ENVÍO DE FICHEROS AL SERVIDOR A TRAVÉS DE UN SERVLET

8. Envío de ficheros al servidor a través de un servlet

Cuando realizamos una aplicación Java, es posible que tengamos la necesidad de crear un **sistema de administración**, normalmente en la Intranet, para simplificar las tareas de mantenimiento. Esta administración posiblemente estará formada por unos cuantos formularios, normalmente interactuará con una base de datos aunque a veces tenemos la necesidad de **subir un fichero al servidor**.

Cuando se quiere implementar el envío de ficheros al servidor a través de un servlet, el formulario debe de ser **enctype="multipart/form-data"** para que admita el envío del fichero. El primer inconveniente que encontramos es que **si el envío es multipart, no funcionan las llamadas con request.getParameter(), siempre devuelven null.** En versiones anteriores a Servlet 3.0 la práctica más común es hacer uso de **Apache Commons FileUpload** para analizar la multipart del formulario de solicitud de datos. Si la API es Servlet 3.0 o superior, entonces se puede utilizar HttpServletRequest para recoger de varias partes del formulario los datos (en la mayoría de Servlet 3.0 sus implementaciones utilizan Apache Commons FileUpload internamente).

Hay que anotar la carga de archivos mediante servlets con la **anotación multipartConfig** para manejar las peticiones **"form-data"** de varias partes del formulario que se va a utilizar para subir el archivo al servidor. La anotación **multipartConfig** tiene los siguientes atributos:

- ✓ **fileSizeThreshold**: nos puede especificar el máximo tamaño a subir. El valor del tamaño en bytes es 1024*1024*10, que son 10MB.
- ✓ **location**: directorio donde los archivos se almacenan de forma predeterminada, el valor por defecto es "".
- ✓ **maxFileSize**: el tamaño máximo permitido para cargar un archivo, que el valor se proporciona en bytes. Su valor por defecto es **1L** y significa ilimitado.
- ✓ **maxRequestSize**: el tamaño máximo permitido para la petición multipart/form-data. El valor por defecto es **1L** también.

Los **métodos de HttpServletRequest** para obtener todas las partes en la solicitud de multipart/form-data son **getParts()** para recuperar todas las partes del formulario o podemos obtener una parte específica usando el método **getPart (String partName).**

8.1 Ejemplo

```
@WebServlet("/ServletFileUpload")
@MultipartConfig(fileSizeThreshold = 1024 * 1024 * 10, // 10 MB
        maxFileSize = 1024 * 1024 * 50, // 50 MB
        maxRequestSize = 1024 * 1024 * 100) // 100 MB

public class ServletFileUpload extends HttpServlet {
    private static final long serialVersionUID = 1L;
    private static final String UPLOAD_DIR = "uploads";

    /**
     * Default constructor.
     */
    public ServletFileUpload() {
        // TODO Auto-generated constructor stub
    }
```

```
    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse
     *      response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        String applicationPath = getServletContext().getRealPath("");
        // Construye la ruta del directorio donde se va a guardar el fichero
        // subido
        String uploadFilePath = applicationPath + UPLOAD_DIR;
        PrintWriter out = response.getWriter();
        // Crear el directorio en el caso de que no exista
        File fileSaveDir = new File(uploadFilePath);
        if (!fileSaveDir.exists()) {
            fileSaveDir.mkdirs();
        }
        out.println("Directorio subida =" + fileSaveDir.getAbsolutePath());

        String fileName = null;
        // Conseguir todas las partes de la request
        for (Part p : request.getParts()) {
            // Recuperar el nombre del fichero a guardar
            if (p.getName().equals("fichero")) {
                fileName = getFileName(p);
                // Guardar en la parte referente al fichero la ruta donde se va a
                // guardar el fichero
                System.out.println(uploadFilePath + File.separator + fileName);
                // Carga el fichero en la ruta indicada
                p.write(uploadFilePath + File.separator + fileName);
                out.println(" Fichero subido correctamente!");
            }
        }

    }
```

```
    private String getFileName(Part part) {
        String fileName = "";
        String contentDisp = part.getHeader("content-disposition");
        System.out.println("content-disposition header= " + contentDisp);
        String[] tokens = contentDisp.split(";");
        for (String token : tokens) {
            if (token.trim().startsWith("filename")) {
                if (token.contains("\\")) {
                    fileName = token.substring(token.lastIndexOf("\\") + 1, token.length() - 1);
                } else {
                    fileName = token.substring(token.lastIndexOf("=") + 1, token.length() - 1);
                }

            }
        }
        return fileName;

    }

}
```