

# ¿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

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

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

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

 Powered by 	Hosting Patrocinado por <b>enREDados.com</b> 
---	--

[Home](#) | [Quienes Somos](#) | [Empleo](#) | [Tutoriales](#) | [Contacte](#)

	<p><b>Tutorial desarrollado por: <a href="#">Carlos García Pérez</a></b></p> <p>Puedes encontrarme en <a href="#">Autentia</a>  <b>Somos expertos en Java/J2EE</b>  <b>Contacta en <a href="mailto:carlos.garcia@autentia.com">carlos.garcia@autentia.com</a></b></p>	<p><a href="http://www.adictosaltrabajo.com">www.adictosaltrabajo.com</a> es el Web de difusión de conocimiento de <a href="http://www.autentia.com">www.autentia.com</a></p>  <p><b>autentia</b>          real business solutions</p> <p><a href="#">Catálogo de cursos</a></p>
---	---	--

Descargar este documento en formato PDF [JakartaCommonsIO.pdf](#)

[Firma en nuestro libro de Visitas](#)

#### [Curso de Desarrollo Java](#)

60 Horas 420 Euros, inicio 20 Feb  
 Matrícula incluida, 915590611  
[www.tecsur.es](http://www.tecsur.es)

#### [Master Java Certificado](#)

Temario Actualizado-UML-JSF-  
 AJAX Trabajo Garantizado-Bolsa  
 de Empleo  
[www.exes.es](http://www.exes.es)

#### [DynamicPDF Generator v4.0](#)

Highly Efficient Java PDF Library.  
 Encryption, Barcodes and HTML  
 Text.  
[www.dynamicpdf.com](http://www.dynamicpdf.com)

#### [Proyectos LabView](#)

¿Necesita desarrollar aplicaciones  
 basadas en LabView?  
[www.adcon.es](http://www.adcon.es)

[Anuncios Google](#)

[Anunciarse en este sitio](#)

# Apache Jakarta Commons IO

## Introducción

En este tutorial vamos a hacer una presentación de la librería Jakarta Commons IO de Apache.

Esta librería nos proporciona una extensión a la funcionalidad de los paquetes de Entrada/Salida que nos proporciona la plataforma J2SE.

Para obtener una información más extensa puede dirigirse a la [web oficial](#).

Commons IO se compone de varios paquetes que nos proporcionan:

- Clases de utilidad
- Endian. (Si has trabajado con sockets entre distintas plataformas estas clases te servirán de gran ayuda)
- Iteradores de líneas para stream de caracteres.
- Nuevos InputStream y OutputStream.

## Ejemplos

### Clases de utilidad

#### [org.apache.commons.io.FileUtils](#)

Esta clase nos proporciona una valiosa funcionalidad para realizar las siguientes tareas:

1. Lectura, escritura, modificación, creación y borrado de ficheros y directorios.
2. Comparación entre ficheros.
3. Facilidades de búsqueda en directorios.
4. Cálculo de Checksums.

En la siguiente aplicación resalto la funcionalidad que a mi personalmente me ha parecido más útil.

```

import java.io.*;
import java.util.*;
import org.apache.commons.io.*;
import org.apache.commons.io.filefilter.*;

/**
 * Varios ejemplos sobre la clase @see org.apache.commons.io.FileUtils
 * @author Carlos García. Autentia
 */
public class FileUtilsExamples {
    /**
     * Punto de inicio del ejemplo
     */
    public static void main(String[]){
        InputStream input1 = null;
        InputStream input2 = null;
        OutputStream output1 = null;
        Reader reader1 = null;
        final File testDir = new File(System.getProperty("java.io.tmpdir"));
        final File testDir2 = new File(System.getProperty("java.home"));
        final File testFile1 = new File("c:/a.txt");
        final File testFile2 = new File("c:/b.txt");

        try {
            // Averiguamos el tamaño total de un directorio.
            System.out.println("Nº bytes del directorio: " + FileUtils.sizeOfDirectory(testDir2));

            // Imprime un tamaño en un formato más legible para las personas.
            System.out.println( FileUtils.byteCountToDisplaySize(length)); // Por ejemplo: 80 MB

            // Elimina el contenido de un directorio sin eliminar el directorio.
            // FileUtils.cleanDirectory(testDir);

            // Elimina un fichero cuando finalice la máquina virtual que ejecuta la aplicación.
            // En caso de especificar un directorio, borra todos los ficheros de todos
            // los subdirectorios, pero sin eliminar los subdirectorios
            // FileUtils.forceDeleteOnExit(testFile);

            // Compara la fecha de creación de dos ficheros
            if (FileUtils.isFileNewer(testFile1, testFile2)){
                System.out.println(testFile1.toString() + " es más reciente");
            } else {
                System.out.println(testFile2.toString() + " es más reciente");
            }

            // Leemos el contenido de un fichero de texto con codificación iso-8859-1
            String content = FileUtils.readFileToString(testFile1, "iso-8859-1");
            System.out.println(content);

            // Obtenemos una colección (de objetos File) de todas las imágenes
            // un directorio y subdirectorios
            String[] exts = new String[] {"jpg", "gif", "png"};
            Collection images = FileUtils.listFiles(testDir2, exts, true);
            System.out.println("Existen " + images.size() + " imágenes");

            // Implementa el comportamiento del comando 'touch' de Unix.
            // a) Si el fichero no existe lo crea con tamaño 0.
            // b) Si el fichero existe actualiza su fecha de modificación a la hora actual
            // sin modificar su contenido.
            FileUtils.touch(testFile2);

            // Escribe una cadena con codificación UTF-8 en un fichero.
            // Si el fichero no existiese lo crea.
            FileUtils.writeStringToFile(testFile2, "Carlos García Pérez", "utf-8");
        } catch (Exception ex){
            System.out.println(ex);
        } finally {
            // Cerramos los Stream de prueba.
            // Nota, podemos olvidándonos de tener que capturar CUALQUIER excepción. ;- )
            IOUtils.closeQuietly(input1);
            IOUtils.closeQuietly(input2);
            IOUtils.closeQuietly(output1);
            IOUtils.closeQuietly(reader1);
        }
    }
}

```

## org.apache.commons.io.IOUtils

Esta clase nos proporciona una valiosa funcionalidad para leer, escribir y comparar Stream, Readers y Writers.

En la siguiente aplicación resalto la funcionalidad que a mi personalmente me ha parecido más útil.

```
import java.io.*;
import java.util.*;
import org.apache.commons.io.*;
import org.apache.commons.io.filefilter.*;

/**
 * Varios ejemplos sobre la clase @see org.apache.commons.io.IOUtils
 * @author Carlos García. Autentia
 */
public class IOUtilsExamples {

    /**
     * Ficheros con los que realizamos las pruebas.
     */
    private static String TestFile1 = "c:/a.txt";
    private static String TestFile2 = "c:/outCommonsIOTest.txt";

    /**
     * Punto de inicio del ejemplo
     */
    public static void main(String[] args){
        InputStream input1 = null;
        InputStream input2 = null;
        OutputStream output1 = null;
        Reader reader1 = null;

        try {

            // Copiamos el contenido de un InputStream en un OutputStream
            // (Todos los métodos están sobrecargados para usar Readers y Writers)
            input1 = new FileInputStream(TestFile1);
            output1 = new FileOutputStream(TestFile2);
            IOUtils.copy(input1, output1);

            // Cerramos los Stream de la prueba
            IOUtils.closeQuietly(input1);
            IOUtils.closeQuietly(output1);

            // Comparamos dos InputStream
            input1 = new FileInputStream(TestFile1);
            input2 = new FileInputStream(TestFile2);

            if (IOUtils.contentEquals(input1, input2)){
                System.out.println("Los ficheros son iguales");
            } else {
                System.out.println("Los ficheros son distintos");
            }

            // Cerramos los ficheros de la prueba. No lanza excepciones aunque sean null.
            IOUtils.closeQuietly(input1);
            IOUtils.closeQuietly(input2);

            // Convertimos una cadena a un byte[] con codificación de caracteres UTF-8
            String name = "Cañón";
            byte[] bytes = IOUtils.toByteArray(new StringReader(name), "UTF-8");
            System.out.println(name.getBytes().length); // => 5
            System.out.println(bytes.length); // => 7

            // Leemos un stream de texto a través de un Iterator
            reader1 = new FileReader(TestFile1);
            LineIterator fileIte = IOUtils.lineIterator(reader1);
            try {
                while (fileIte.hasNext()) {
                    String line = fileIte.nextLine();

                    // Realizamos el tratamiento de la línea
                    System.out.println(line);
                }
            } finally {
                fileIte.close();
                IOUtils.closeQuietly(reader1);
            }

            // Averiguamos el número de líneas de un Stream de texto.
            reader1 = new FileReader(TestFile1);
            java.util.List list1 = IOUtils.readLines(reader1);
            IOUtils.closeQuietly(reader1);
            System.out.println("El fichero tiene " + list1.size() + " líneas.");

            // Leemos un recurso a través de su URL. En este caso una página web
            try {
                input1 = new java.net.URL("http://www.google.es").openStream();
                System.out.println(IOUtils.toString(input1));
            } finally {
                IOUtils.closeQuietly(input1);
            }
        }
    }
}
```

```

        } catch (Exception ex){
            System.out.println(ex);
        } finally {
            // Cerramos los Stream de prueba.
            // Nota, podemos olvidándonos de tener que capturar CUALQUIER excepción. ;-))
            IOUtils.closeQuietly(input1);
            IOUtils.closeQuietly(input2);
            IOUtils.closeQuietly(output1);
            IOUtils.closeQuietly(reader1);
        }
    } // FIN main
} // FIN class

```

## org.apache.commons.io.FilenameUtils

Esta clase nos proporciona una valiosa funcionalidad para el tratamiento de rutas de archivos.

En la siguiente aplicación resalto la funcionalidad que a mi personalmente me ha parecido más útil.

```

import java.io.*;
import java.util.*;
import org.apache.commons.io.*;
import org.apache.commons.io.filefilter.*;

/**
 * Varios ejemplos sobre la clase @see org.apache.commons.io.FilenameUtils
 * @author Carlos García. Autentia
 */
public class FilenameUtilsExamples {
    /**
     * Punto de inicio del ejemplo
     */
    public static void main(String[] args){
        System.out.println(FilenameUtils.concat("c:/windows", "c:/pepe/a")); // c:\pepe\a
        System.out.println(FilenameUtils.concat("c:/windows", "pepe/a")); // c:\windows\pepe\a
        System.out.println(FilenameUtils.normalize("c:/windows/./")); // c:\windows\
        System.out.println(FilenameUtils.getBaseName("c:/windows/a.bmp")); // a
        System.out.println(FilenameUtils.getExtension("c:/windows/a.bmp")); // bmp
        System.out.println(FilenameUtils.getFullPath("c:/windows/a.bmp")); // c:/windows/
        System.out.println(FilenameUtils.getPath("c:/windows/a.bmp")); // windows/
        System.out.println(FilenameUtils.separatorsToSystem("c:/windows/a.bmp")); // c:\windows\
        System.out.println(FilenameUtils.wildcardMatch("a.bmp", "*.bmp")); // true
        System.out.println(FilenameUtils.wildcardMatch("a.bmp", "*.jpg")); // false
        System.out.println(FilenameUtils.wildcardMatch("a/b/a.bmp", "a/b/*")); // true
        System.out.println(FilenameUtils.wildcardMatch("a.bmp", ".*???")); // true
        System.out.println(FilenameUtils.wildcardMatch("a.bmp", ".*????")); // false
    }
}

```

## org.apache.commons.io.filefilters.\*

Esta clase nos proporciona una valiosa funcionalidad para realizar búsquedas parametrizadas sobre archivos.

En la siguiente aplicación os muestro como utilizar y combinar filtros. Deseo resaltar que existen muchos filtros predefinidos más.

```

import java.io.*;
import java.util.*;
import org.apache.commons.io.*;
import org.apache.commons.io.filefilter.*;

/**
 * Varios ejemplos sobre las clases del paquete @see org.apache.commons.io.filefilters
 * @author Carlos García. Autentia
 */
public class FileFiltersExamples {
    /**
     * Punto de inicio del ejemplo
     */
    public static void main(String[] args){
        File testDir = new File(System.getProperty("java.io.tmpdir"));
        String[] files = null;
        IOFileFilter filter = null;
        String[] cads = null;

        // Mostramos los archivos del directorio que comienzan por 'a' o 'A'
        cads = new String[] {"a", "A"};
        filter = new PrefixFileFilter(cads);
        files = testDir.list(filter);
        for (int i = 0, nFiles = files.length; i < nFiles; i++) {
            System.out.println(files[i]);
        }

        // Combinamos dos filtros para mostramos las imágenes de más de 10Kb.
        cads = new String[] {".jpg", ".png", ".gif", ".bmp"};
        filter = new AndFileFilter(new SizeFileFilter(1024 * 10), new SuffixFileFilter(cads));
        files = testDir.list(filter);
        for (int i = 0, nFiles = files.length; i < nFiles; i++) {
            System.out.println(files[i]);
        }
    }
}

```

## Conclusiones

Como veis este tutorial no tiene una elevada complejidad ni explica tecnologías complejas, pero creo que este API es interesante y puede ahorrar tiempo en el desarrollo de cualquier proyecto.

Bueno, espero que os haya sido de utilidad este tutorial.

En Autentia Real Business Solutions, nos gusta compartir el conocimiento. Aquí teneis un poquito más de nuestra aportación.

Si algún día necesitais ayuda con vuestros proyectos o necesitais formación, podéis encontrarlos en [Autentia](#)



This work is licensed under a [Creative Commons Attribution-NonCommercial-No Derivative Works 2.5 License](#).  
[Puedes opinar sobre este tutorial aquí](#)



## Recuerda

que el personal de [Autentia](#) te regala la mayoría del conocimiento aquí compartido ([Ver todos los tutoriales](#))

¿Nos vas a tener en cuenta cuando necesites consultoría o formación en tu empresa?

**¿Vas a ser tan generoso con nosotros como lo tratamos de ser con vosotros?**

[info@autentia.com](mailto:info@autentia.com)

Somos pocos, somos buenos, estamos motivados y nos gusta lo que hacemos .....

**Autentia = Soporte a Desarrollo & Formación**

### Formación en nuevas tecnologías

[Autentia S.L.](#) Somos expertos en:  
**J2EE, Struts, JSF, C++, OOP, UML, UP, Patrones de diseño ..**  
 y muchas otras cosas

## 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	<input type="text"/>
	<input type="button" value="Enviar"/>

## Otros Tutoriales Recomendados ([También ver todos](#))

### Nombre Corto

[Apache Commons Configuration](#)

[Apache, MySQL y PHP](#)

[mod\\_jk en Linux / Apache2-JBoss](#)

[mod\\_jk en WindowsXP / Apache2-JBoss](#)

[Activación de la seguridad en Apache](#)

[Apache Commons Lang](#)

[Leer un documento de Microsoft Word usando la librería POI de jakarta](#)

### Descripción

En este tutorial os vamos a enseñar a utilizar una API de Apache para gestionar las configuraciones de vuestras aplicaciones de manera avanzada

Os mostramos como configurar Apache, MySQL y PHP en vuestra máquina

El conector mod\_jk se encarga de enviar las peticiones dinámicas de Apache2 a un servidor de aplicaciones JBoss

Os mostramos como instalar el conector mod\_jk sobre WindowsXP utilizando Apache2 y JBoss

Alejandro Pérez nos enseña como securizar Apache a través de autenticación básica y certificados de seguridad SSL.

En este tutorial se verá un ejemplo que incluyen clases y métodos interesantes.

En este documento aprenderemos a leer un documento de Microsoft Word usando la librería POI de jakarta

- [mod\\_jk en Ubuntu / Apache2-JBoss](#) Os mostramos como instalar el conector mod\_jk sobre la distribución linux Ubuntu utilizando Apache2 y JBoss
- [Manual Básico de Apache iBatis](#) En este tutorial aprenderemos el uso básico de iBatis
- [Jakarta Commons HttpClient](#) Este tutorial pretende que mediante un ejemplo podamos entender el funcionamiento de una comunicación HTTP, usando el framework HttpClient del lado del cliente

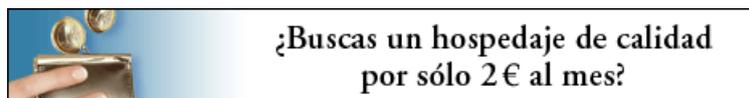
Nota: Los tutoriales mostrados en este Web tienen como objetivo la difusión del conocimiento.

Los contenidos y comentarios de los tutoriales son responsabilidad de sus respectivos autores.

En algún caso se puede hacer referencia a marcas o nombres cuya propiedad y derechos es de sus respectivos dueños. Si algún afectado desea que incorporemos alguna reseña específica, no tiene más que solicitarlo.

Si alguien encuentra algún problema con la información publicada en este Web, rogamos que informe al administrador rcanales@adictosaltrabajo.com para su resolución.

[Patrocinados por enredados.com .... Hosting en Castellano con soporte Java/J2EE](#)



www.AdictosAlTrabajo.com Opimizado 800X600