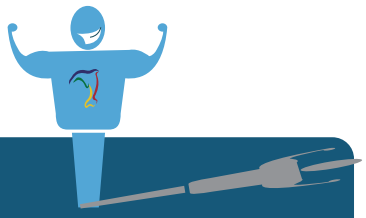
autentia

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...

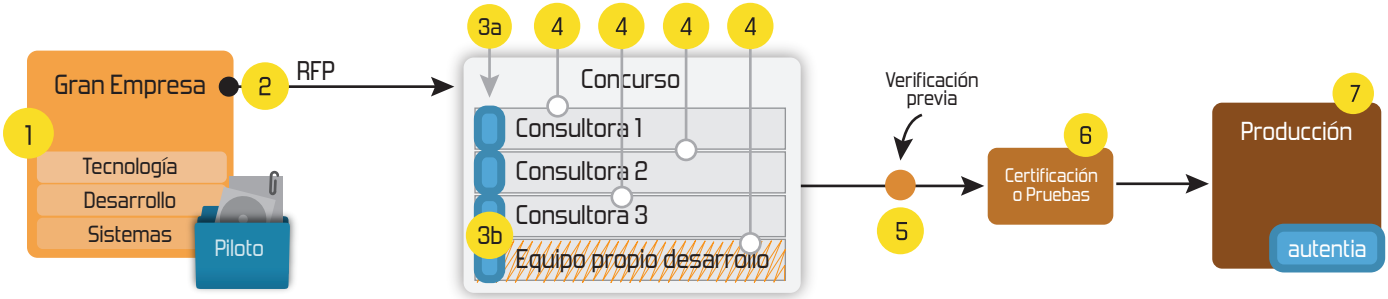
1. Desarrollo de componentes y proyectos a medida



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

Respuesta serializada entre Applets y Servlets

En uno de nuestros tutoriales anteriores, os mostramos como se podría comunicar un Applet y un servlet.

Dentro de los procedimientos de comunicación, una de las técnicas preferidas es la serialización de datos. En el citado tutorial os mostrabamos una porción de código para comunicar un applet con un servlet y os comentamos que la comunicación al contrario sería igual.

Ya he recibido varios correos donde os interesáis como hacerlo ... podéis creerme.... es igual (aunque hay que tener en cuenta reglas simple) .... y para demostrarlo, os voy a poner un ejemplo.

Obviamente, si la respuesta que vas a retornar la vas a querer procesar como un elemento binario... debes eliminar cualquier respuesta textual ... o asegurarte de ignorarla en el Stream de lectura del applet..

Podemos ir al grano y ver el servlet que hemos generado

```
/*
 * servletsimple.java
 *
 * Modificado el 15 abril de 2004
 */

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

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

/**
 *
 * @author  Roberto Canales
 * @version
 */
public class servletsimple extends HttpServlet {

   protected void processRequest(HttpServletRequest request, HttpServletResponse response)
   throws ServletException, IOException {

      // establecemos el formato de la respuesta
      response.setContentType("application/octet-stream");

      try
      {
         // recuperamos el stram de entrada
         ObjectInputStream bufferentrada = new ObjectInputStream(request.getInputStream());

         // leemos un array de datos
```

```
            int[] arrayRecuperado = (int[])bufferentrada.readObject();

            // cramos una variable auxiliar
            String cadenaAux = "";

            // iteramos por los elementos
            for(int i=0; i < arrayRecuperado.length ;i++)
            {
               System.out.println("El valor recuperado del elemento " + i + " es " + arrayRecuperado[i]);

               // concatenamos los elementos que leemos
               cadenaAux += (arrayRecuperado[i] + " - ");
            }

            // Configurarmos un Stream de Salida
            ObjectOutputStream buffersalida = new ObjectOutputStream(response.getOutputStream());

            // construimos el objeto a retornar
            String[] resultado = new String[2];

            // asignamos un elemento fijo
            resultado[0]="Esta es la cadena retornada";

            // y concatenamos el variable
            resultado[1]= cadenaAux;

            // escribimos los datos
            buffersalida.writeObject(resultado);

            // y los enviamos
            buffersalida.flush();
         }
          catch(Exception e)
          {
             System.out.println("Error al recuperar datos");
          }
       }

      protected void doGet(HttpServletRequest request, HttpServletResponse response)
      throws ServletException, IOException {
         processRequest(request, response);
      }

      protected void doPost(HttpServletRequest request, HttpServletResponse response)
      throws ServletException, IOException {
         processRequest(request, response);
      }
}
```

Ahora vamos a realizar una pequeña modificación del applet para que al pulsar el botón.... se envíe un array de numeros y podeamos leer el array de cadenas que nos retorne el sistema.

No tenemos porque enviar o recibir un solo elemento .... auque para demostrar que funciona es suficiente.

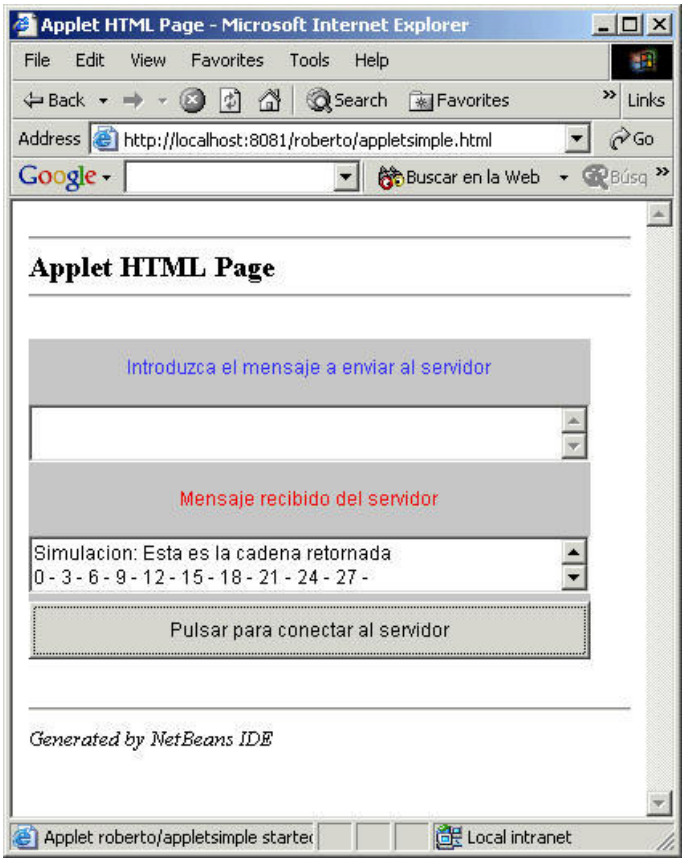
Vemos ahora el código del Applet

```
/*
 * appletsimple.java
 *
 * Modificado el 15 de Abril de 2004
 */

package roberto;

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

/**
 *
 * @author  Roberto Canales
 */
public class appletsimple extends java.applet.Applet {

   /** Initializes the applet appletsimple */
   public void init() {
      initComponents();
   }

   /** This method is called from within the init() method to
    * initialize the form.
    * WARNING: Do NOT modify this code. The content of this method is
    * always regenerated by the Form Editor.
    */
   private void initComponents() {
      label1 = new java.awt.Label();
      mensajeenviar = new java.awt.TextArea();
      label2 = new java.awt.Label();
      mensajerecibir = new java.awt.TextArea();
      botonenviar = new java.awt.Button();

      setLayout(new java.awt.GridLayout(5, 1, 2, 5));
```

```
    label1.setAlignment(java.awt.Label.CENTER);
    label1.setForeground(new java.awt.Color(51, 51, 255));
    label1.setText("Introduzca el mensaje a enviar al servidor");
    add(label1);

    add(mensajeenviar);

    label2.setAlignment(java.awt.Label.CENTER);
    label2.setForeground(new java.awt.Color(255, 0, 0));
    label2.setText("Mensaje recibido del servidor");
    add(label2);

    add(mensajerecibir);

    botonenviar.setLabel("Pulsar para conectar al servidor");
    botonenviar.addActionListener(new java.awt.event.ActionListener() {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            botonenviarActionPerformed(evt);
        }
    });

    add(botonenviar);

}

private void botonenviarActionPerformed(java.awt.event.ActionEvent evt) {

    String s_textoaenviar = mensajeenviar.getText();
    mensajerecibir.setText("Simulacion: " +  peticionPost(s_textoaenviar));

}

// enviamos la peticion por post
String peticionPost(String mensaje) {
    URL miurl = null;
    String cadenaaux = null;
    String cadenaretorno = "";

    String consulta = "param1=" + URLEncoder.encode(mensaje);

    try {

        // recuperamos la direccion de servidor destino
        String host = this.getCodeBase().getHost();

        // elegimos la URI del recurso a invocar
        String peticion = "/servlet/servletsimple";

        // montamos la direccion base
        miurl = new URL(getCodeBase(),peticion);

        // abrimos la conexion
        URLConnection conexion = miurl.openConnection();

        // aactivamos la salida
        conexion.setDoOutput(true);

        // Creamos el stream de escritura
        ObjectOutputStream buffersalida = new ObjectOutputStream(conexion.getOutputStream());

        // montamos un array de elementos para enviarlos
        int array[] = new int[10];

        // inicializamos el array
        for(int i=0;i <array.length;i++)
        {
            array[i] = i*3;
        }

        // escribimos el objeto
        buffersalida.writeObject(array);

        // enviamos
        buffersalida.flush();

        // Creamos el Stream de lectura
        ObjectInputStream bufferentrada = new ObjectInputStream(conexion.getInputStream());

        // creamos la referencia que recogera los datos
        String respuesta[];

        // leemos la respuesta
        respuesta = (String [])bufferentrada.readObject();

        // iteramos por la respuesta y montamos el mensaje
        for(int i=0 ; i < respuesta.length; i++)
        {
            cadenaretorno += (respuesta[i] + "\n");
        }

        buffersalida.close();
        bufferentrada.close();
    }
    catch (Exception e)
    {
        return "Error al generar url " + e.getMessage();
```

```
        }

        return cadenaretorno;
    }


    // enviamos por get la petición
    String peticionGet(String mensaje) {
        URL miurl = null;
        String cadenaaux = null;
        String cadenaretorno = "";

        try {
            String host = this.getCodeBase().getHost();
            String peticion = "/servlet/servletsimple?param1=" + URLEncoder.encode(mensaje);
            miurl = new URL(getCodeBase(),peticion);
            InputStream buffer = miurl.openStream();
            BufferedReader bufferreader = new BufferedReader(new InputStreamReader(buffer));

            while( (cadenaaux = bufferreader.readLine()) != null) {
                cadenaretorno += cadenaaux;
            }

            buffer.close();
        }
        catch (Exception e) {
            return "Error al generar url " + e.getMessage();
        }

        return cadenaretorno;
    }


    // Variables declaration - do not modify
    private java.awt.Button botonenviar;
    private java.awt.Label label1;
    private java.awt.Label label2;
    private java.awt.TextArea mensajeenviar;
    private java.awt.TextArea mensajerecibir;
    // End of variables declaration

}
```

Y, activando el ejemplo .... vemos que funciona correctamente. ¿A que no era tan complicado ;-) ?



Podéis bajaos el WAR (Web ARchive)... y recordar que es un ZIP renombrado (poco más o menos)

Cuando he tenido la necesidad de realizar estos procesos de comunicación (applet/servlet), he procurado que sea completamente transparente el mecanismo de transporte: Sockets, cadenas textuales, RMI, serialización, etc... Esto es fácil si tenéis conocimientos básicos de patrones de diseño.

En concreto, una implementación sencilla del patrón estrategia (polimorfismo puro) junto con una factoría .... puede solucionar el problema ... aunque esto... es otra historia....

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
Transformación de XML y XSL en JSPs	Os mostramos como poder utilizar XML y XSL en JSPS, combinado con el Patrón MVC
Generar imagenes desde Servlets	Os mostramos como generar ficheros GIF desde un servlet java. Util para generear gráficas dinámicas, contadores, etc
Uso de JNDI, includes y cookies en Servlets	En este tutorial veremos como usar variables de entorno desde JNDI, incluir un servlet en otro (include) y como usar cookies en Servlets
Upload de ficheros en Java	Os mostramos como enviar ficheros a un servidor Web y manipularlos en un servlet en el servidor, gracias a APIs de apache
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
Novedades en Java 1.5	Ya está disponible la versión Beta del J2SDK 1.5. Os mostramos algunas de las nuevas características introducidas en el lenguaje Java: Clases genéricas, enumeraciones, bucles simplificados, etc.
Struts Jakarta	Cuando se ha trabajado creando aplicaciones Java poco a poco se va viendo la necesidad de normalizar los desarrollo. Uno de los Framework (entornos) más extendidos es Struts
JSP´s y Modelo-Vista-Controlador	En este tutorial os enseñamos como crear un JSP, su relación con los servlets y como crear un ejemplo MVC en Tomcat
Optimización de Serialización Java	Os mostramos una sencilla técnica para mejorar el rendimiento de la serialización de objetos en Java, a través de Streams asociados a buffers en memoria.

www.AdictosAlTrabajo.com Opimizado 800X600