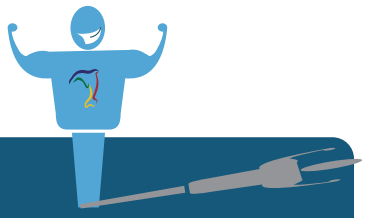
autentia

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...

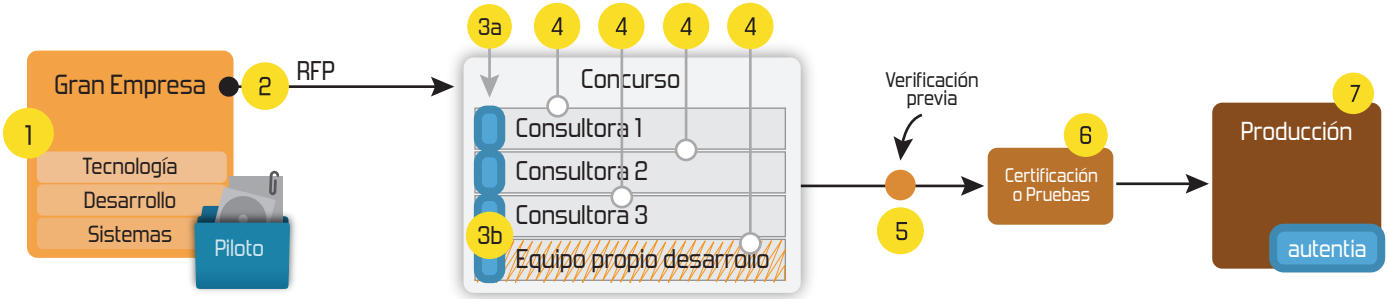
1. Desarrollo de componentes y proyectos a medida



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 conociemiento en:
www.adictosaltrabajo.com

Para más información visítenos en:
www.autentia.com

Descargar este documento en formato PDF appletservlet.pdf

Comunicación entre Applets y Servlets

Nos ha realizado muchas consultas sobre la comunicación entre applets y servlets.

Un applet y un servlet se pueden comunicar de muchos modos (sockets, RMI, corba, etc) aunque el modo más sencillo de hacerlo es a través de peticiones HTTP.

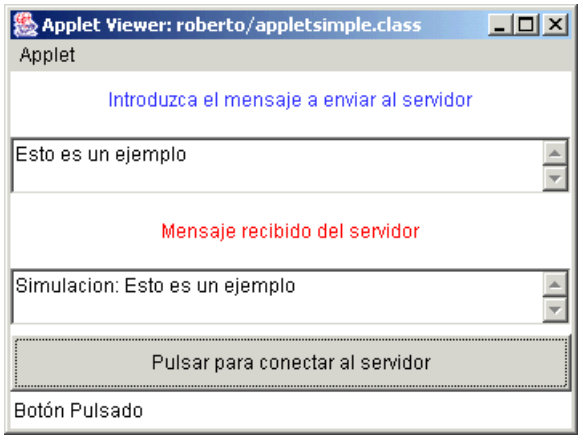
Preparación

Primero, vamos a crear un applet básico. Este applet lo hemos creado a través de NetBeans por lo que podéis encontrar algunas partes del código o comentarios en inglés.

```
/*
 * appletsimple.java
 *
 * Created on November 5, 2003, 10:51 PM
 */

package roberto;

/**
 *
 * @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: " + s_textoaenviar);
         this.showStatus("Botón Pulsado");
    }


    // 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

}
```

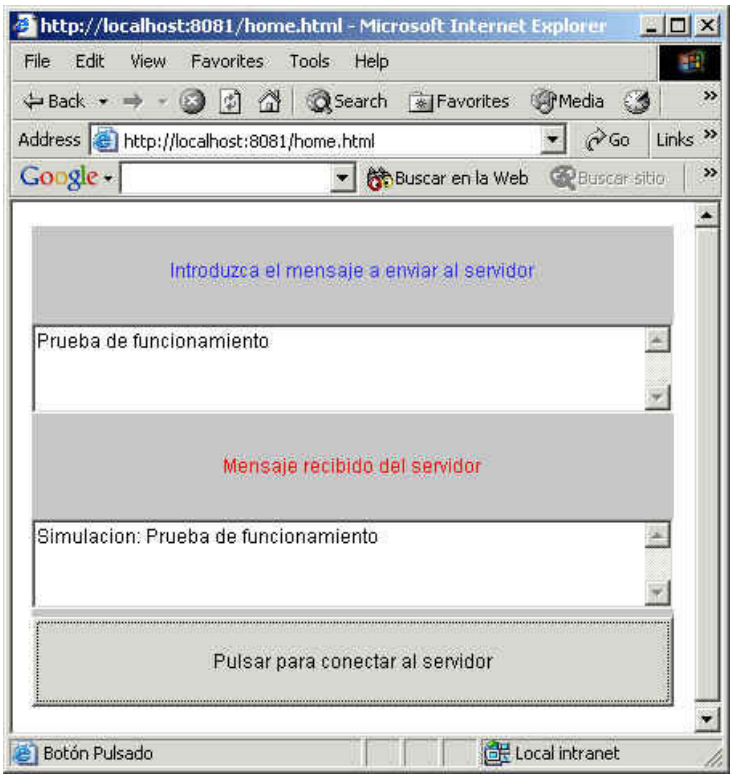
Como podéis observar, en depuración, el código anterior lo único que hace es mostrar un pequeña ventana donde al introducir un texto en la parte superior y pulsar el botón, muestra un texto en la parte de inferior y un mensaje en la barra de estado.



Para poder probar nuestro programa de un modo efectivo, debemos crear una página HTML y colgarla de un servidor Web.

```
<html>
<body>
<p align="center">
<applet width="400" height="300" code="roberto.appletsimple.class" codebase="./">Applet que se comunica con servlet</applet>
</p>

</body>
</html>
```

El aspecto de nuestra página en el navegador es:



Ahora, vamos a crear el servlet que se ejecutará en el servidor. Este servlet solamente se encarga de recoger los parámetro que recibe en la petición Get o Post y genera una página de respuesta mostrándolos.

```
/*
* servletsimple.java
*
* Created on November 5, 2003, 11:38 PM
*/

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 {
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>");

Enumeration e = request.getParameterNames();

while(e.hasMoreElements())
{
String s_clave = e.nextElement().toString();
String s_aux = request.getParameter(s_clave);
out.println("<br>El parámetro: <B>" + s_clave + "</B> vale: <B>" + s_aux + "</B>");
}

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

out.close();
}

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);
```

```
}
}
```

Podemos ver la petición y la respuesta en la siguiente pantalla.

Insertar código comunicaciones

Ahora lo único que nos queda en modificar el código del applet para hacer la petición HTTP al servidor. Esta versión del código os muestra como hacerlo a través de POST pero podéis ver que también está dentro del código la gunción para hacerlo a través del comando GET.

peticionPost(s_textoaenviar)

Vamos a comentar las funciones más significativas (por ejemplo con la fucnión GET):

Primero, recuperamos el Host del que descargamos el Applet

String host = this.getCodeBase().getHost();

Posteriormente, debemos codificar los caracteres especiales al juego compatible con HTTP ()

String peticion = "/servlet/servletsimple?param1=" + URLEncoder.encode(mensaje);

Generamos la URL que representa el recurso a recuperar (en este caso el servlet)

miurl = new URL(getCodeBase(),peticion);

Ahora conectamos con el recurso

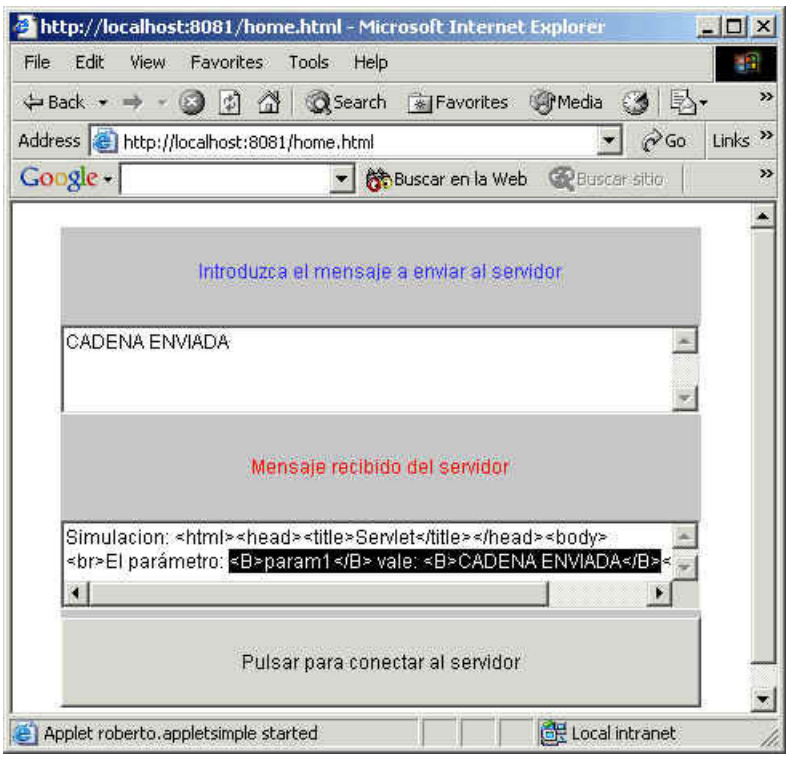
InputStream buffer = miurl.openStream();

```
/*
 * appletsimple.java
 *
 * Created on November 5, 2003, 10:51 PM
 */

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 {
        String host = this.getCodeBase().getHost();
        String peticion = "/servlet/servletsimple";
        miurl = new URL(getCodeBase(),peticion);

        URLConnection conexion = miurl.openConnection();
        conexion.setDoOutput(true);
        OutputStreamWriter buffersalida = new OutputStreamWriter(conexion.getOutputStream());
        buffersalida.write(consulta);
        buffersalida.flush();

        BufferedReader bufferentrada = new BufferedReader(new InputStreamReader(conexion.getInputStream()));
        String linea = null;

        while ((linea = bufferentrada.readLine()) != null) {
            cadenaretorno += linea;
        }

        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 el resultado es



Sencillo verdad .....

Serialización de comunicaciones

Una vez que sabemos hacer esto, podemos utilizar una petición POST de un modo un poquito especial .....

serializando la llamada entre el cliente y el servidor y de este modo, enviar y recibir objetos completos.

Hacemos un pequeño cambio, primero en el Applet, donde vamos a enviar un array

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

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

   try {
   String host = this.getCodeBase().getHost();
   String peticion = "/servlet/servletsimple";
   miurl = new URL(getCodeBase(),peticion);

   URLConnection conexion = miurl.openConnection();
   conexion.setDoOutput(true);
   ObjectOutputStream buffersalida = new ObjectOutputStream(conexion.getOutputStream());
   //buffersalida.write(consulta);

   int array[] = new int[10];
   for(int i=0;i<array.length;i++)
   {
      array[i] = i*3;
   }

   buffersalida.writeObject(array);

   buffersalida.flush();

   BufferedReader bufferentrada = new BufferedReader(new  InputStreamReader(conexion.getInputStream()));
   String linea = null;
```

```
  while ((linea = bufferentrada.readLine()) != null) {
    cadenaretorno += linea;
    }

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

  return cadenaretorno;

}
```

Y ahora cambiamos el servlet

```
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>");

/*
Enumeration e = request.getParameterNames();

while(e.hasMoreElements())
{
String s_clave = e.nextElement().toString();
String s_aux = request.getParameter(s_clave);
out.println("<br>El parámetro: <B>" + s_clave + "</B> vale: <B>" + s_aux + "</B>");
}
*/

  try
  {
    ObjectInputStream bufferentrada = new ObjectInputStream(request.getInputStream());

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

    for(int i=0;i<arrayRecuperado.length;i++)
    {
       out.println("El valor recuperado del elemento " + i + " es " + arrayRecuperado[i]);
    }
  }
  catch(Exception e)
  {
    out.println("Error al recuperar datos");
  }

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

  out.close();
}
```

Y en el sentido contrario..... es exactamente lo mismo .....

Esto es impresionante..... imaginaos lo rápido que es el desarrollo. Además, cuando nos cuentan sobre nuevas tecnologías de interacción Web, como Web Services ..... tampoco nos asusta tanto

Proxys y autentificación

Por cierto, si tenéis **proxy** ..... hay que establecer unas propiedades

```
System.setProperty("proxyHost", "host");

System.setProperty("proxyPort", "puerto");
```

Y si tenéis seguridad (**autentificación simple**) en el servidor, debéis añadir unas líneas a vuestro objeto **URLConnection** :

```
conexion.setRequestProperty( "Authorization","BASIC " + passwordencriptadaenbase64);
conexion.setRequestProperty("content-type", "text/html");
```

De todos modos, debéis tener en cuenta donde usar los applets .... Yo os recomendaría restringirlos a Intranets.

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
Firma digital de un Applet	Para que un applet Java pueda ejecutarse en un cliente Web con la configuración de seguridad por defecto y/o adquirir privilegios de seguridad, es necesario firmarla digitalmente. Alejandro Perez nos enseña como hacerlo de un modo rápido y sencillo.
Cachear porciones de JSPs	En este tutorial os ensañamos como incrementar increiblemente el rendimiento de vuestro Web basado en tecnología JSP con el FrameWork de cache OSCACHE
Aplicación básica con RMI	Gracias a este tutorial, podreis aprender paso a paso como crear una aplicación cliente-servidor con RMI
Construir un Servidor Web en Java	En este tutorial os enseñamos los principios de las aplicaciones multi-hilo a través de la creación de un servidor web básico en Java. Podremos ver en un ejemplo real el uso de sockets, threads, excepciones, etc.
Control navegación en Servlets	Os mostramos como construir el esqueleto de una aplicación basada en Servlets y JSP, con control de navegación.
Técnicas básicas y poco comentadas en Java	Os mostramos como realizar algunas cosas simples en Java: Formateo de decimales y enteros, gestión de preferencias y comparación entre objetos de nuevas clases
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
Applet con gráficas JFreeChart	Os mostramos como mostrar en un applet las graficas generadas por JFreeChart sin necesidad cargar las clases en el cliente

www.AdictosAlTrabajo.com Opimizado 800X600