



Laboratorio de Aplicaciones Telemáticas Ingeniería Técnica de Telecomunicación Especialidad en Telemática

Escuela Politécnica Superior. Universidad Carlos III de Madrid.
Leganés, a 9 de Febrero de 2004.

Duración de la prueba: 2h.

Lea las preguntas con atención y sea concreto en sus respuestas. Si es necesario realizar alguna suposición, indíquelo claramente.

La empresa Hoza y Goza S.A. ha registrado hyg.com, una web de cocina tradicional, a través de la cual va a ofrecer un servicio (mediante JSP y servlets) de planificación de comidas para sus usuarios. El servicio dará soporte a los procesos diarios de toma de decisión del tipo ¿qué vamos a comer hoy?, ¿qué puedo cocinar con lo que tengo en la nevera?, o ¿no comimos carne ayer? HyG ofrece recetas alternativas dependiendo de los conocimientos culinarios del usuario y de los ingredientes de que dispone. HyG clasifica los conocimientos de cocina en tres niveles (básico, avanzado y grand chef), y también ofrece tres cursos a distancia con streaming MPEG de audio y video, para ilustrar la preparación de las recetas.

Los usuarios acceden a las aplicaciones de sugerencia de recetas y de teleeducación a través de las urls <http://hyg.com/sugerencias> y <http://hyg.com/cursos> respectivamente. En ambas aplicaciones es necesario autenticarse como paso inicial, y proporcionar al sistema el identificador del usuario, la contraseña y el nivel de conocimientos culinarios.

1. Escriba el formulario (no es necesario escribir la página XHTML completa) para enviar los datos de autenticación al servlet `login` de la aplicación de sugerencias. de forma que viajen codificados en la URI. Indique el método y la codificación HTTP de la petición. (1 punto)

Respuesta:

```
<form action=/sugerencias/login method=get>
Identificador: <input type=text name=login id=login />
Contraseña: <input type=password name=passwd id=passwd />
Conocimientos: <select name=conocimientos>
<option label="basico" value=0>básico</option>
<option label="avanzado" value=1>avanzado</option>
<option label="gran chef" value=2>gran chef</option>
</select>
<input type=submit value=enviar />
</form>
GET /sugerencias/login?login=nombre&
passwd=xx&conocimientos=1 HTTP/1.1
```

El funcionamiento de la aplicación de sugerencia de recetas está basado en un árbol de decisión que recorren los usuarios. Cada nodo ofrece una alternativa para cada rama que sale de él, por ejemplo el nodo raíz ofrece las alternativas: **carne, pescado, pasta, verduras y ensaladas**. Los nodos hoja contienen las recetas a sugerir. Se está pensando en implementar el árbol a través de objetos java, como se ilustra en el cuadro 1.

2. Sabiendo que el árbol de decisión es común para todos los usuarios y que contiene 10.000 nodos de los cuales 5.500 son hojas, razone qué ámbito (*scope*) tendría la variable `árbol`, y cuándo se construiría (quién la invocaría y desde qué método). (1 punto)

Respuesta: tendría ámbito de aplicación (*context*), pues es común a todas las sesiones de todos los usuarios, y se metería en el contexto utilizando `ServletContext.setAttribute()`. La forma más sencilla de construirlo es sobrescribiendo el método `GenericServlet.init()`.

3. ¿Qué variables es necesario almacenar en la sesión? (1 punto).

Respuesta: por lo menos una referencia al objeto nodo en el que se encuentra el usuario, y su nivel de conocimientos.

4. Enumere las distintas formas que conoce de implementar sesiones y explique las con una frase. (1 punto)

Respuesta: campos ocultos en formulario...
reescritura de URLs...
cookies...

5. Programe el código del método `doPost` del servlet `Selecciona` cuyo cometido actual es recibir la respuesta del usuario, seleccionar el siguiente nodo en función del nodo actual y la respuesta elegida, construir una página xhtml con un formulario con las nuevas alternativas, y si se alcanza una hoja, pasar la explicación al JSP `MiReceta`. (2 puntos)

Respuesta:

```
public doPost (HttpServletRequest req, HttpServletResponse res)
throws ServletException{
    HttpSession ses= req.getSession();
    Nodo nodo= ses.getAttribute("nodo");
    String e= req.getHeader("respuesta");
    Nodo sig= nodo.getHijo(e);
    if (sig!=null){
        ses.setAttribute("nodo", sig);
        if (sig.isHoja()){
            String con= ses.getAttribute("conocimientos");
            req.setAttribute("receta", sig.getReceta(con));
            getServletContext("getRequestDispatcher("MiReceta.jsp").\
                forward(req,res);
```

```

    }
    res.setContentType("text/html");
    out= res.getWriter();
    out.println("<?xml version=\"1.0\" encoding=\"ISO-8859-1\" ?>");
    ...
    out.println("<form action=Selecciona method=POST>");
    String[] rr= sig.getRespuestas();
    for (int i=0; i<rr.length; i++)
        out.println("<input type=radio name=respuesta"
                    value=\""+rr[i]+\"/>" + rr[i]);
    ...
    }
    throw new ServletException("Respuesta incorrecta");
}

```

HyG quiere ofrecer un nuevo servicio de acceso a las recetas en formato multimedia con un sistema de prepago, a los usuarios de la aplicación de sugerencia de recetas. Para ello quiere reutilizar el servlet `MandaMpeg` que necesita el identificador de la receta, desde el servlet `Selecciona`.

6. Escriba el código que tendría que añadirse a `Selecciona`. Nota: tenga en cuenta que los servlets se encuentren en distintas aplicaciones. (1 punto)

Respuesta:

```

req.setAttribute("id", sig.getID());
ServletContext sc=
Servlet.getServletConfig().getServletContext().\
getServletContext("/cursos");
sc.getRequestDispatcher("MandaMpeg").forward(req,res);

```

o bien

```

req.setAttribute("id", sig.getID());
res.sendRedirect("/cursos/MandaMpeg");

```

HyG ha decidido comprar un software que permite recomendar a sus usuarios más asiduos un sistema de régimen individualizado, combinado con un sistema de compra de víveres e ingredientes culinarios. Para ello el sistema almacena históricos de la interacción de dichos usuarios con el sistema. Tras la implantación de la nueva aplicación, se constata que el número de usuarios que demandan el servicio es tan alto que es necesaria una fuerte inversión en nuevos servidores. La dirección estratégica recomienda empaquetar la aplicación para su ejecución en local y pasar a la venta directa.

Tras un estudio minucioso se opta por diseñar un interfaz con `java.swing` y reaprovechar todo lo posible el sistema existente.

7. Programe la versión Swing de la clase `NavegaSwing` que permite la navegación del usuario a través del árbol de sugerencias. Tenga en cuenta que el usuario en cada momento está en un nodo del árbol, que se proporcionan al usuario tantos botones como alternativas ofrece el nodo, y que al ser pulsados la acción será la misma (cambiar al nodo seleccionado). Suponga que la clase `NavegaSwing` extiende `JFrame` y tiene un constructor con un parámetro (el nodo inicial). (1.5 puntos)

Respuesta:

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class NavegaSwing extends JFrame implements ActionListener {

    Nodo nodo;
    String[] resp;
    JButton[] botones;
    JLabel[] etiquetas;
    ContentPane cp;
    public NavegaSwing (Nodo n, int depth) {
        super();
        nodo = n;
        resp= nodo.getRespuestas();
        botones = new JButton[depth];
        cp= getContentPane();
        cp.setLayout(new GridLayout());
    }

    public void mostrarAlternativas() {
        for (int i=0; i < resp.length; i++) {
            if (botones[i] != null){
                botones[i].setText(resp[i]);
                botones[i].setVisible(true);
            }
            else{
                botones[i]= new JButton(resp[i]);
                cp.add(botones[i]);
            }
        }
    }

    public void ocultarAlternativas() {
        for (int i=0; i < resp.length; i++)
            botones[i].setVisible(false);
    }
}
```

```

public void actionPerformed(ActionEvent e) {
    Nodo sig = nodo.selecciona(e.getActionCommand());
    if (sig != null && !sig.isHoja()) {
        ocultarAlternativas();
        nodo = sig;
    }
    resp= nodo.getRespuestas();
    mostrarAlternativas();
    pack();
    setVisible(true);
}

public static void main(String[] args) {
    int depth= 50; // Máxima profundidad
    Nodo root= construir_árbol();

    NavegaSwing navega = new NavegaSwing(root, depth);
    navega.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    navega.mostrarAlternativas();
    navega.pack();
    navega.setVisible(true);
}
}

```

El departamento de atención al cliente de HyG constata que el 85% de las sugerencias piden algún método alternativo a la tediosa tarea de recorrer algunos nodos en los que se pregunta si el usuario tiene determinados ingredientes (como aceite, sal o vinagre) que podría inferirse que no se han gastado desde la última vez que se utilizó el sistema. HyG decide ofrecer un servicio individualizado de estimación de inventario, de forma que se evite este problema.

8. Para la funcionalidad de estimación de inventario es necesario llevar la cuenta de las alternativas seleccionadas por el usuario. Esta operación debe ser realizada sin modificar el manejador de eventos del apartado anterior. ¿Es esto posible? ¿Cómo? (0.5 puntos)

Respuesta: Si, asociando un `ActionListener` adicional a cada botón con un método `ActionPerformed` que lleve la cuenta de cada opción seleccionada.

9. Explique el modelo de eventos en java: su ciclo de vida y los interfaces y clases relevantes. Limite su respuesta a los botones y los cuadros de texto. (1 punto)

```

public class Nodo{
    Nodo[] hijos= null;
    String[] respuestas= null;
    int opciones= 0;

    public boolean isHoja(){return false};
    public void setRespuestas(int n){
        opciones= n;
        hijos= new Nodo[n];
        respuestas= new String[n];
    }
    public void setHijo(int i, String s){
        respuestas[i]= new String(s);
        hijos[i]= new Nodo();
    }
    public String[] getRespuestas(){
        return respuestas;
    }
    public Nodo selecciona(String r){
        for (int i= 0; i<opciones; i++)
            if (respuestas[i].equalsIgnoreCase(r))
                return hijos[i];
        return null;
    }
}

public class Hoja extends Nodo{
    Receta receta;

    public boolean isHoja(){return true};
    public setReceta(Receta r){
        receta= r;
    }
    public String getReceta(int conocimientos){
        return r.explicar(conocimientos);
    }
    public int getId(){
        return r.getId();
    }
}

```

Cuadro 1: Implementación del árbol en objetos