[Tarea 8. Desarrollo de un cliente para un servicio web REST](https://m4gm.com/moodle/mod/assign/view.php?id=1861)

Principio del formulario

Final del formulario

Cada alumno deberá desarrollar un programa Java consola (modo carácter) cliente del servicio web que creamos en la tarea 7.

Se deberá realizar las siguientes modificaciones al servicio web que creamos en la tarea 7:

1. Agregar el campo id\_usuario a la clase Usuario.

2. Modificar el método web "alta\_usuario", de manera que al dar de alta un usuario el método web deberá regresar al cliente el id del usuario agregado. Se deberá desplegar el id del usuario dado de alta. El campo id\_usaurio es auto\_increment en la base de datos, por tanto se deberá recuperar el ID inmediatamente después de ejecutar la instrucción INSERT.

3. Modificar el método web "consulta\_usuario", ahora la consulta se deberá realizar mediante el id del usuario no el email.

4. Modificar el método web "modifica\_usuario", utilizar el id del usuario en el WHERE de la instrucción UPDATE en lugar del email. No deberá ser posible modificar el id de un usuario ya que se trata de la llave primaria.

5. Modificar el método web "borra\_usuario", utilizando como clave el id del usuario no el email.

El programa cliente deberá desplegar el siguiente menú:

MENU

a. Alta usuario  
b. Consulta usuario

c. Borra usuario

d. Salir

Opción: \_

Las opciones deberán implementar la siguiente funcionalidad:

* La opción "Alta usuario" leerá del teclado el email, el nombre del usuario, el apellido paterno, el apellido materno, la fecha de nacimiento, el teléfono y el género ("M" o "F"). Entonces se invocará el método web "alta\_usuario". Se deberá desplegar el id del usuario dado de alta, o bien, el mensaje de error que regresa el servicio web. Notar que el método web "alta\_usuario" recibe como parámetro una instancia de la clase Usuario, recordemos que esta clase se deberá definir de la siguiente manera:

    class Usuario  
    {

      id\_usuario int,  
      String email;  
      String nombre;  
      String apellido\_paterno;  
      String apellido\_materno;  
      String fecha\_nacimiento;  
      String telefono;  
      String genero;  
      byte[] foto;  
    }

    Para invocar el método web "alta\_usuario" desde el cliente Java es necesario crear una instancia de la clase Usuario y asignar los valores a los campos (en este caso el campo "foto" será null). Una vez que se tenga el objeto de tipo Usuario se deberá utilizar **GSON** para convertir el objeto a una string JSON, entonces se deberá utilizar está string como valor del parámetro (ver más adelante cómo se enviarán los parámetros a los métodos web).

* La opción "Consulta usuario" leerá del teclado el id de un usuario previamente dado de alta. Entonces se invocará el método web "consulta\_usuario". Si el usuario existe, se desplegará en pantalla el nombre del usuario, el apellido paterno, el apellido materno, la fecha de nacimiento, el teléfono y el género. Debido a que el programa no es gráfico, la foto del usuario se ignorará. Notar que el método web "consulta\_usuario" regresa una string JSON la cual representa un objeto de tipo Usuario, por tanto será necesario utilizar **GSON** para convertir la string JSON a un objeto Java de tipo Usuario y posteriormente desplegar los campos del objeto (excepto el campo "foto"). Si hubo error, se desplegará el mensaje que regresa el servicio web.
* La opción "Borra usuario" leerá del teclado el id de un usuario previamente dado de alta. Entonces se invocará el método "borra\_usuario" del servicio web. Se deberá desplegar "El usuario ha sido borrado" si se pudo borrar el usuario, o bien, el mensaje de error que regresa el servicio web.
* La opción "Salir" terminará el programa.

**¿Cómo invocar un método web REST utilizando Java?**

A continuación se muestra un ejemplo de código Java que permite invocar el método web "consulta\_usuario" del servicio web **Servicio.war**

La URL que utilizaremos para acceder el método web "consulta\_usuario" es la siguiente (notar que el nombre del método se escribe al final de la URL):

http://*ip-de-la-máquina-virtual*:8080/Servicio/rest/ws/consulta\_usuario

Para que el método web pueda recibir los parámetros, cada parámetro se codifica de la siguiente manera: **nombre=valor**. Los parámetros se deben separar con un caracter **&**

El **valor** debe codificarse utilizando el método URLEncoder.encode(). La codificación URL reemplaza los caracteres reservados por una secuencia de escape consistente en un % y un valor hexadecimal, ver: <https://developers.google.com/maps/documentation/urls/url-encoding>

Por ejemplo, si un método web tiene tres parámetrosa, b y c, y los parámetros tienen los valores 10, 20 y 30, entonces se deberá enviar al método web los parámetros de la siguiente manera: "a=10&b=20&c=30", en este caso no es necesario utilizar el método URLEncoder.encode() para codificar el valor ya que se trata de un número.

El código que permite invocar el método "consulta\_usuario" del servicio web Servicio.war es el siguiente:

URL url = new URL("http://*ip-de-la-máquina-virtual*:8080/Servicio/rest/ws/consulta\_usuario");

HttpURLConnection conexion = (HttpURLConnection) url.openConnection();

conexion.setDoOutput(true);

// en este caso utilizamos el método POST de HTTP  
conexion.setRequestMethod("POST");

// indica que la petición estará codificada como URL  
conexion.setRequestProperty("Content-Type","application/x-www-form-urlencoded");

// el método web "consulta\_usuario" recibe como parámetro el id de un usuario, en este caso el id es 10  
String parametros = "id=" + URLEncoder.encode("10","UTF-8");

OutputStream os = conexion.getOutputStream();

os.write(parametros.getBytes());

os.flush();

// se debe verificar si hubo error  
if (conexion.getResponseCode() == 200)  
{  
 // no hubo error

BufferedReader br = new BufferedReader(new InputStreamReader((conexion.getInputStream())));

String respuesta;

// el método web regresa una string en formato JSON  
 while ((respuesta = br.readLine()) != null) System.out.println(respuesta);

}  
else  
{  
 // hubo error  
 BufferedReader br = new BufferedReader(new InputStreamReader((conexion.getErrorStream())));  
  
 String respuesta;  
  
 // el método web regresa una instancia de la clase Error en formato JSON  
 while ((respuesta = br.readLine()) != null) System.out.println(respuesta);  
  
 // dispara una excepción para terminar el programa  
 throw new RuntimeException("Codigo de error HTTP: " + conexion.getResponseCode());  
}  
  
conexion.disconnect();

Notar que el código anterior se puede usar para desarrollar aplicaciones Android que requieran consumir servicios web REST.

Se deberá entregar un reporte en formato PDF que incluya las capturas de pantalla donde se muestre cada paso, desde la creación de la máquina virtual hasta la compilación y ejecución de cada opción del menú. El reporte deberá tener portada y conclusiones. Se deberá entregar también el todos los archivos utilizados incluyendo el código fuente del servicio web y el cliente.

El nombre de la máquina virtual deberá ser el número de boleta del alumno, si el número de boleta del alumno es 12345678, entonces la máquina virtual deberá llamarse: X12345678. **No se admitirá la tarea** si la máquina virtual no se nombra como se indicó anteriormente.

Las capturas de pantalla deberán ser completas, no se aceptará la tarea si incluye recortes de capturas de pantalla.

Valor de la tarea: 30% (1.8 puntos de la segunda evaluación parcial)