GUIA DESPLIEGUE MTIS

Javier Molpeceres Gómez

Contenido

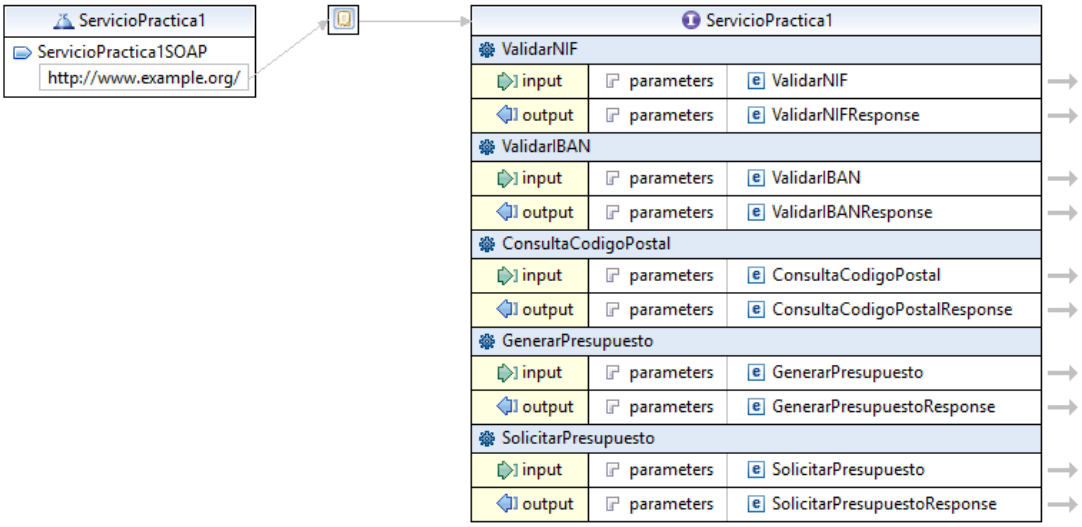
Introducción

A continuación, se explicará en que consiste la práctica 1 de MTIS. Se nos pide realizar un servicio web mediante el enfoque Top-Down, mediante eclipse. También se nos pide realizar un cliente .NET para consumir este servicio.

Cabe destacar que este servicio necesita de la validación de una clave para consumir los servicios, así cada servicio recibe una variable con la clave, se comprueba y si es correcta se realiza el servicio, si no se devuelve un mensaje de error.

Servicio Web

Explicaremos brevemente el servicio web realizado en Eclipse. Este servicio web creado mediante eclipse nos genera mediante el esquema del **wsdl** las clases necesarias.



Las partes desarrolladas se han realizado en **ServicioPractica1Skeleton** a parte se han desarrollado distintas clases para realizar las comprobaciones, por ejemplo, TrabajarBD que realiza las acciones sobre la Base de datos, las clases iniciadas con la etiqueta Respuesta realizan las acciones.

Este servicio web ofrece los siguientes métodos:

- **ValidarNIF**: recibirá un NIF el cual recibe un NIF y se comprueba si la combinación de números obtiene la letra dada. De la siguiente manera:

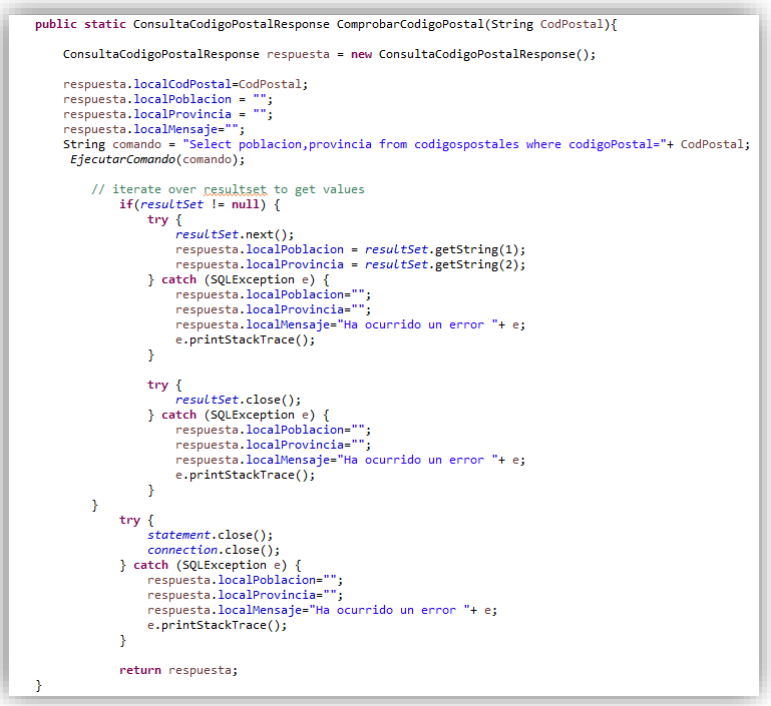
```
public ValidarNIFResponse RealizarValidacionNIF(){
    String juegoCaracteres="TRWAGMYFPDXBNJZSQVHLCKE";
    Respuesta.localValidado = false;

    if(ValidacionNif.localNIF.length()==9){
        int dni = Integer.parseInt(ValidacionNif.localNIF.substring(0,8));
        char letra= ValidacionNif.localNIF.charAt(8);
        int modulo= dni % 23;
        if(letra == juegoCaracteres.charAt(modulo)){
        Respuesta.localValidado=true;
        }
    }


    return Respuesta;
}
```

- **validarIBAN:** recibirá un numero de IBAN y validará de la siguiente manera si es correcto o no:

```
public static Boolean destriparIban(String iban) throws Exception {
       if (iban.length() == 24) {
           return validarCuentaES(iban.substring(4, 24));
       } else {
           return false;
       }
   }

   public static Boolean validarCuentaES(String ccc) throws Exception {
       try {
           String banco = ccc.substring(0, 4);
           String oficina = ccc.substring(4, 8);
           String dc = ccc.substring(8, 10);
           String cuenta = ccc.substring(10, 20);
           if (!dc.equals(obtenerDC(banco, oficina, cuenta))) {
               return false;
           } else if ("0000".equals(banco) || "0000".equals(oficina) || "0000000000".equals(cuenta)) {
               return false;
           } else {
               return true;
           }
       } catch (StringIndexOutOfBoundsException e) {
           System.err.println(e.getMessage());
           return false;
       }
   }

   public static String ObtenerDC(String banco, String oficina, String cuenta) throws StringIndexOutOfBoundsException {
       int temp;
       String dg1, dg2;
       temp = 0;

       temp += Integer.parseInt(banco.substring(0, 1)) * 4;
       temp += Integer.parseInt(banco.substring(1, 2)) * 8;
       temp += Integer.parseInt(banco.substring(2, 3)) * 5;
       temp += Integer.parseInt(banco.substring(3, 4)) * 10;
       temp += Integer.parseInt(oficina.substring(0, 1)) * 9;
```

```
    temp += Integer.parseInt(oficina.substring(1, 2)) * 7;
    temp += Integer.parseInt(oficina.substring(2, 3)) * 3;
    temp += Integer.parseInt(oficina.substring(3, 4)) * 6;

    temp = 11 - temp % 11;
    switch (temp) {
        case 10:
            dg1 = String.valueOf(1);
            break;
        case 11:
            dg1 = String.valueOf(0);
            break;
        default:
            dg1 = String.valueOf(temp);
            break;
    }
    temp = 0;

    temp += Integer.parseInt(cuenta.substring(0, 1)) * 1;
    temp += Integer.parseInt(cuenta.substring(1, 2)) * 2;
    temp += Integer.parseInt(cuenta.substring(2, 3)) * 4;
    temp += Integer.parseInt(cuenta.substring(3, 4)) * 8;
    temp += Integer.parseInt(cuenta.substring(4, 5)) * 5;
    temp += Integer.parseInt(cuenta.substring(5, 6)) * 10;
    temp += Integer.parseInt(cuenta.substring(6, 7)) * 9;
    temp += Integer.parseInt(cuenta.substring(7, 8)) * 7;
    temp += Integer.parseInt(cuenta.substring(8, 9)) * 3;
    temp += Integer.parseInt(cuenta.substring(9, 10)) * 6;

    temp = 11 - temp % 11;
    switch (temp) {
        case 10:
            dg2 = String.valueOf(1);
            break;
        case 11:
            dg2 = String.valueOf(0);
            break;
        default:
            dg2 = String.valueOf(temp);
            break;
    }
    return String.valueOf(dg1) + String.valueOf(dg2);
}
```

- **consultaCodigoPostal:** recibe un código postal y devolverá el código postal, la población y la provincia consultándolo en la base de datos.



```
public static ConsultaCodigoPostalResponse ComprobarCodigoPostal(String CodPostal){

    ConsultaCodigoPostalResponse respuesta = new ConsultaCodigoPostalResponse();

    respuesta.localCodPostal=CodPostal;
    respuesta.localPoblacion = "";
    respuesta.localProvincia = "";
    respuesta.localMensaje="";
    String comando = "Select poblacion,provincia from codigospostales where codigoPostal="+ CodPostal;
     EjecutarComando(comando);

        // iterate over resultset to get values
            if(resultSet != null) {
                try {
                    resultSet.next();
                    respuesta.localPoblacion = resultSet.getString(1);
                    respuesta.localProvincia = resultSet.getString(2);
                } catch (SQLException e) {
                    respuesta.localPoblacion="";
                    respuesta.localProvincia="";
                    respuesta.localMensaje="Ha ocurrido un error "+ e;
                    e.printStackTrace();
                }

                try {
                    resultSet.close();
                } catch (SQLException e) {
                    respuesta.localPoblacion="";
                    respuesta.localProvincia="";
                    respuesta.localMensaje="Ha ocurrido un error "+ e;
                    e.printStackTrace();
                }
        }
            try {
                statement.close();
                connection.close();
            } catch (SQLException e) {
                respuesta.localPoblacion="";
                respuesta.localProvincia="";
                respuesta.localMensaje="Ha ocurrido un error "+ e;
                e.printStackTrace();
            }

            return respuesta;
}
```

- **generarPresupuesto:** se le pasa una estructura (fechaPresupuesto, idCliente, referenciaProducto, cantidadProducto) y devolverá la id del siguiente elemento al almacenado en la BD.

```
public static GenerarPresupuestoResponse GenerarPresupuestoBD(GenerarPresupuesto presupuesto){

    GenerarPresupuestoResponse respuesta = new GenerarPresupuestoResponse();
    java.text.SimpleDateFormat sdf =
            new java.text.SimpleDateFormat("yyyy-MM-dd");
    String currentTime = sdf.format(presupuesto.localFechaPresupuesto);
    String comando = "Insert into presupuestos(idCliente, referenciaProducto, cantidadProducto, fechaPresupuesto) values("
    + presupuesto.localIdCliente+" ,'"+presupuesto.localReferenciaProducto +"','"+presupuesto.localCantidadProducto+"','"+currentTime+"');";
    respuesta.localMensaje="";
    EjecutarComandoInsert(comando);

    if(resultadoInsert != 0){
        String comando2 = "Select id from presupuestos where referenciaProducto='"+ presupuesto.localReferenciaProducto+ "' and idCliente="+ presupuesto.localIdCliente
        EjecutarComando(comando2);

        if(resultSet != null){
            try{
                resultSet.next();
                respuesta.localIdPresupuesto= Integer.parseInt(resultSet.getString(1)) + 1;
                respuesta.localPresupuestoGeneradoCorrectamente=true;
            }catch(Exception e){
                respuesta.localIdPresupuesto=0;
                respuesta.localPresupuestoGeneradoCorrectamente=false;
                respuesta.localMensaje="Ha ocurrido un error "+ e;
            }

            try {
                resultSet.close();
            } catch (SQLException e) {
                respuesta.localIdPresupuesto=0;
                respuesta.localPresupuestoGeneradoCorrectamente=false;
                respuesta.localMensaje="Ha ocurrido un error "+ e;
                e.printStackTrace();
            }

        }else{
            respuesta.localIdPresupuesto=0;
            respuesta.localPresupuestoGeneradoCorrectamente=false;
        }
    }

    try {
        statement.close();
        connection.close();
    } catch (SQLException e) {
        respuesta.localIdPresupuesto=0;
        respuesta.localPresupuestoGeneradoCorrectamente=false;
        e.printStackTrace();
    }

    return respuesta;
}
```

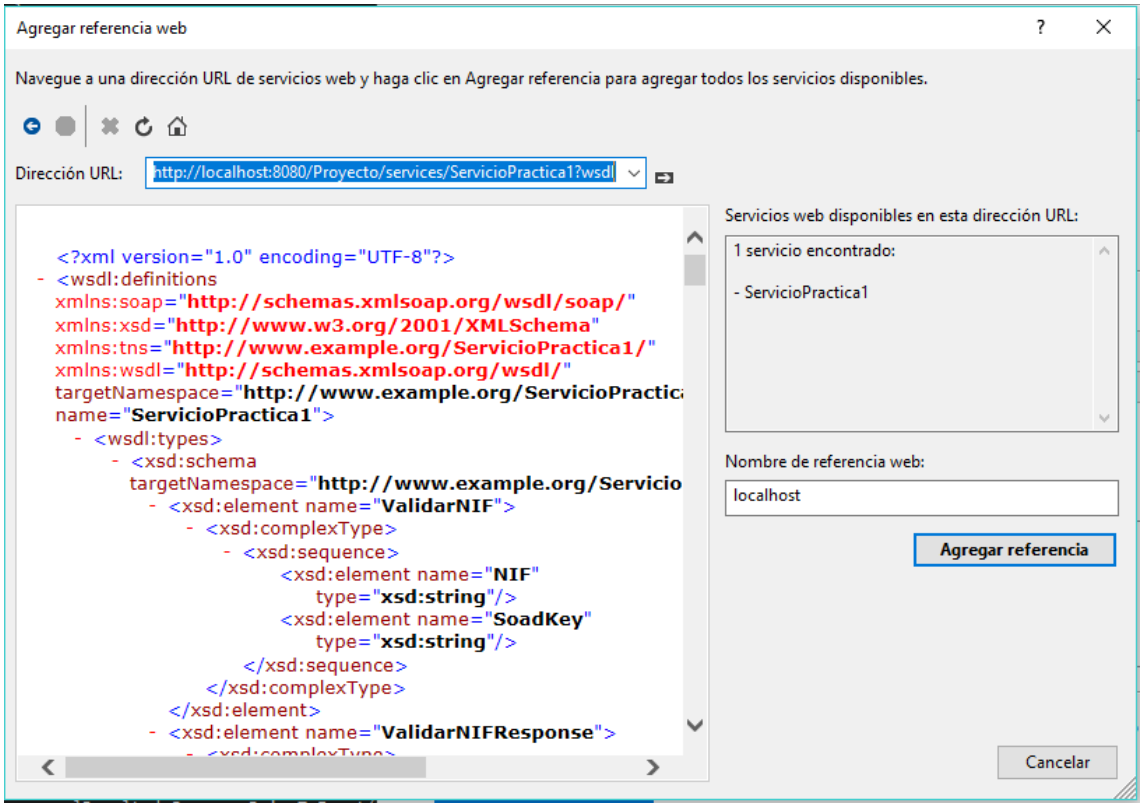
- **solicitarPresupuesto:** se le pasa una estructura (referenciaPrecia, idProveedor) y devuelve el precio de la pieza la disponibilidad y la fecha de disponibilidad.

```
public static SolicitarPresupuestoResponse SolicitarPresupuestoBD(SolicitarPresupuesto presupuesto){
    SolicitarPresupuestoResponse respuesta = new SolicitarPresupuestoResponse();

    respuesta.localMensaje="";
    respuesta.localDisponibilidadPieza=false;
    respuesta.localFechaDisponibilidadPieza=null;
    respuesta.localPrecioPieza=0;
    String comando = "Select precioPieza,disponibilidadPieza,fechaDisponibilidad from piezas where referenciaPieza= '"+ presupuesto.localReferenciaPieza + "' and idProv
    EjecutarComando(comando);

    if(resultSet!=null){
        try {
            resultSet.next();
            respuesta.localPrecioPieza = resultSet.getInt(1);
            respuesta.localDisponibilidadPieza = resultSet.getBoolean(2);
            respuesta.localFechaDisponibilidadPieza = resultSet.getDate(3);

        } catch (SQLException e) {
            respuesta.localMensaje="Ha ocurrido un error "+e;
            e.printStackTrace();
        }

        try {
            resultSet.close();
        } catch (SQLException e) {
            respuesta.localMensaje="Ha ocurrido un error "+ e;
            e.printStackTrace();
        }

    }

    try {
        statement.close();
        connection.close();
    } catch (SQLException e) {
        respuesta.localMensaje="Ha ocurrido un error "+ e;
        e.printStackTrace();
    }
    return respuesta;
}
```

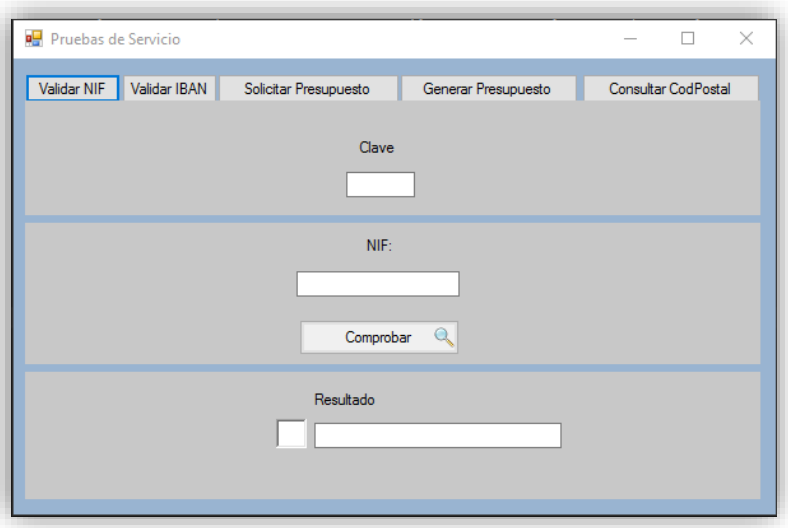
Cliente .NET

Hablaremos a continuación del cliente .NET, este cliente consume el servicio web y ofrece una interfaz sencilla para consumir este servicio.

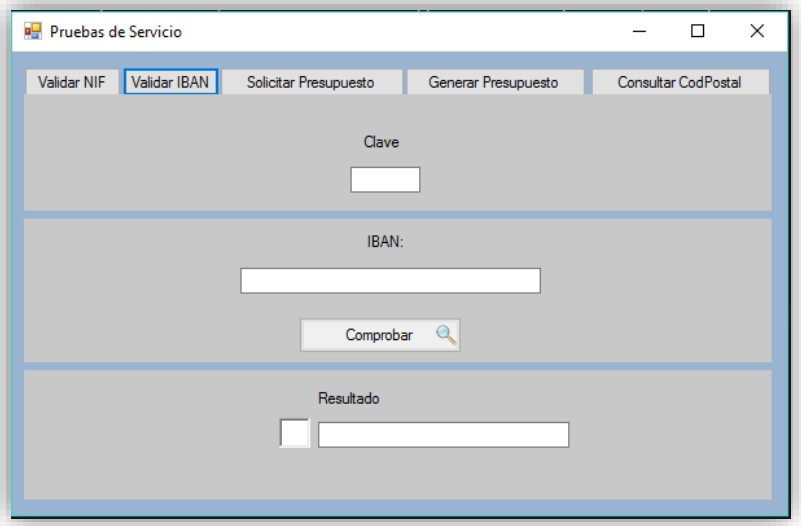
Para poder consumir el servicio en el proyecto ha sido necesario incluir la referencia al servicio web.



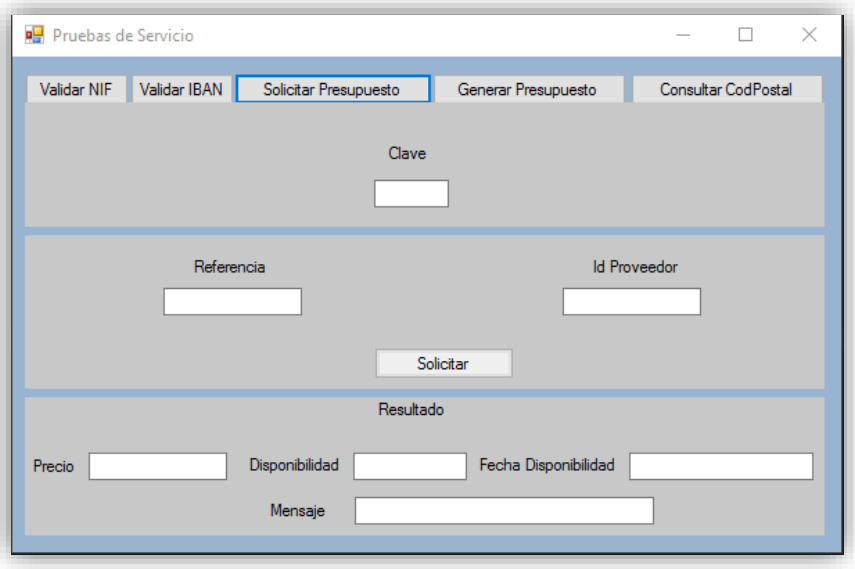
Validación NIF



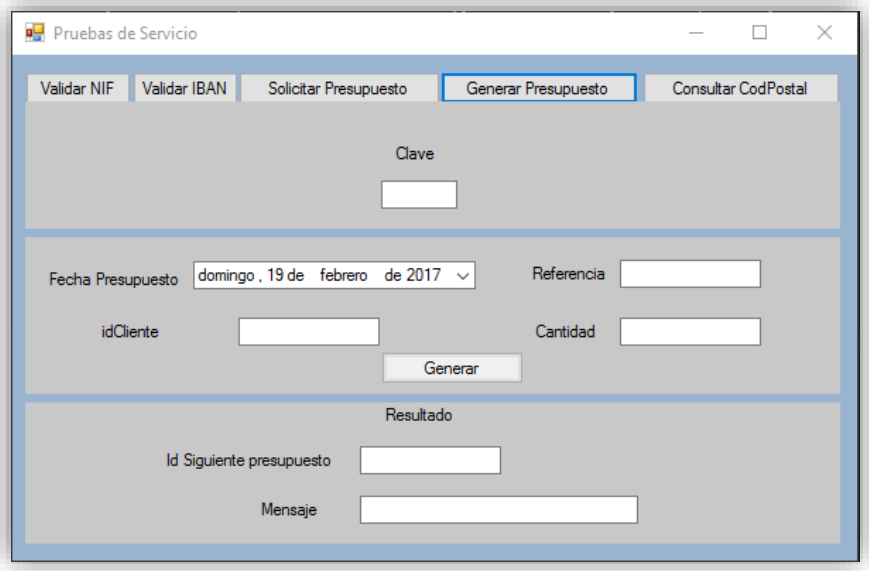
Validación IBAN



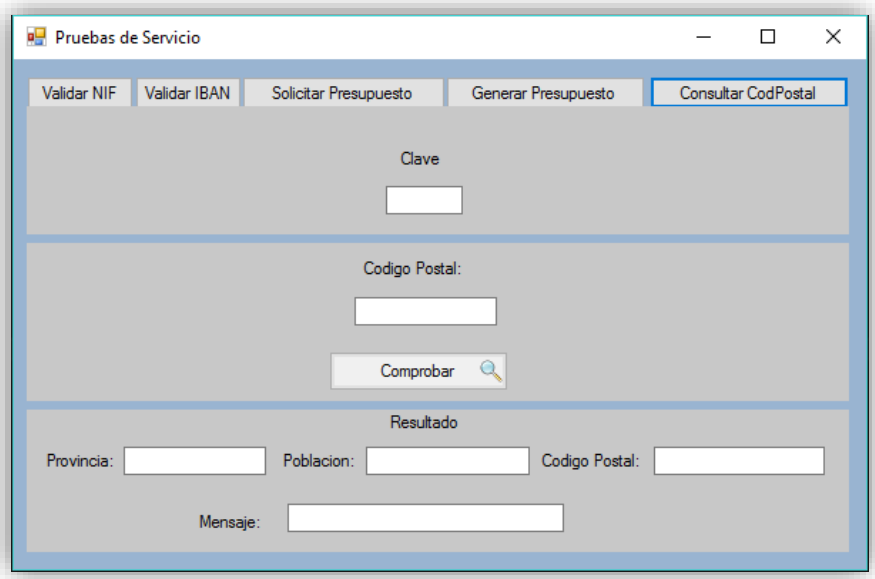
Solicitar Presupuesto



Generar Presupuesto



Consultar Código Postal



Despliegue

Para poder probar la práctica será necesario realizar los siguientes pasos.

1. En primer lugar, arrancaremos la aplicación XAMP con los servicios Apache y Mysql, si no tenemos la base de datos iniciada con datos será necesario ejecutar el script de Base de datos adjuntado en la práctica.

2. El siguiente paso a realizar será arrancar el servicio web de eclipse, sobre la carpeta del proyecto de eclipse pulsaremos en correr en el servidor de apache creado. (Si no tenemos un servidor de apache será necesario crear uno).

3. Por último, iniciaremos la aplicación cliente pulsando sobre **MtisPractica1Cliente.exe(en la carpeta bin/Debug)** y podremos consumir la aplicación.

4. La clave por defecto para poder utilizar el servicio es "*clave*".