|  |  |  |  |
| --- | --- | --- | --- |
|  | | Guía del Web Service del Subsistema de Información sobre Calidad del Aire | |
|  | | [Esta foto](http://poetadelalba1985.blogspot.com/2009/11/conversaciones-con-el-aire.html) de Autor desconocido está bajo licencia [CC BY-NC-ND](https://creativecommons.org/licenses/by-nc-nd/2.5/) | |
| 28/08/2017 | | GUÍA DEL WEB SERVICE | |
|  | Manual de uso del servicio web de AIRE.  IDEAM http://www.siac.gov.co/documents/670372/43391551/02-sisaire.png/81a3b255-7c3d-4e1e-a719-c1017d66c2dd?t=1504017413487 | |

|  |  |  |  |
| --- | --- | --- | --- |
| Versión | Fecha | Elaboró | Visto bueno |
| 1.0 | 2017-08-27 | Santiago Hernández [santiago.h.plazas@gmail.com](mailto:santiago.h.plazas@gmail.com)  Pilar Galindo  [pgalindo@ideam.gov.co](mailto:pgalindo@ideam.gov.co)  Julian Rodriguez [lihernandez@ideam.gov.co](mailto:lihernandez@ideam.gov.co) | Oficina de informática del IDEAM  Subdirección de Estudios Ambientales  Subdirección de ecosistemas e información ambiental |

Contenido

[Introducción 3](#_Toc491973575)

[¿Qué es SISAIRE y su servicio web? 3](#_Toc491973576)

[Propósito del servicio WEB de AIRE 4](#_Toc491973577)

[ACERCA DE ESTA GUIA 4](#_Toc491973578)

[Responsabilidades Básicas del IDEAM y de la Autoridad Ambiental para adelantar la Integración 5](#_Toc491973579)

[Marco Técnico 6](#_Toc491973580)

[Estándares de Interoperabilidad 6](#_Toc491973581)

[Arquitectura Orientada al Servicio (SOA) 6](#_Toc491973582)

[Servicios WEB y el protocolo SOAP 6](#_Toc491973583)

[Ambiente de Pruebas de Consumo del Web Service 8](#_Toc491973584)

[URLs del ambiente de pruebas 8](#_Toc491973585)

[Consumo del Web Service 9](#_Toc491973586)

[Descripción del anexo técnico de AIRE 9](#_Toc491973587)

[Descripción del servicio, reglas del negocio, y detalle técnico de la base de datos 9](#_Toc491973588)

[Diccionario de datos 9](#_Toc491973589)

[Especificacion de mensajes de requerimiento al servicio web 10](#_Toc491973590)

[Lista de autoridades ambientales 10](#_Toc491973591)

[Descripción de mensajes de respuesta 11](#_Toc491973592)

[Preparación de la información en el Cliente 12](#_Toc491973593)

[Sincronización de las listas de valores 12](#_Toc491973594)

[Envío de Reportes 12](#_Toc491973595)

[Implementación del Cliente del Web Service 13](#_Toc491973596)

[Generación de Código Fuente del Cliente del WEB Service usando Netbeans 13](#_Toc491973597)

[Obtener Netbeans 13](#_Toc491973598)

[Ejemplo: registrar la calidad del aire 14](#_Toc491973599)

[Descripción de SoapUI como herramienta de pruebas de los servicios web 41](#_Toc491973600)

[Descarga de SoapUI 41](#_Toc491973601)

[Instalación de SoapUI 41](#_Toc491973602)

[Creación de un proyecto en SoapUI 41](#_Toc491973603)

[Prueba del servicio WEB 44](#_Toc491973604)

[Interpretación del Mensaje de Respuesta 46](#_Toc491973605)

[Verificación de la Información Reportada 46](#_Toc491973606)

# Introducción

## ¿Qué es SISAIRE y su servicio web?

El Sistema de Información sobre Calidad del Aire es un sistema para la captura, almacenamiento, transferencia, procesamiento y consulta de información. También permite la generación de información unificada de las redes de calidad del aire del país.

El SISAIRE permite:

* Recolectar información actualizada y analizada sobre calidad del aire, generada por los sistemas de Vigilancia de Calidad del Aire que son operados por las autoridades ambientales regionales y urbanas, con el propósito de garantizar la disponibilidad y la calidad de la información ambiental y su consulta por los usuarios del sistema.
* Mantener la información al alcance de los ciudadanos y de las instituciones encargadas de la investigación en el tema ambiental.
* Consultar el reporte diario de las mediciones de las Autoridades Ambientales Regionales y de Grandes Centros Urbanos de Colombia que cuentan con Sistemas de Vigilancia de la Calidad del Aire.
* Analizar el consolidado histórico de estas mediciones.
* Estudiar índices e indicadores (Índice Nacional de Calidad del Aire, Estados de Prevención, Alerta y Emergencia y Excedencias de las normas nacionales y regionales de calidad del aire)
* Conocer toda la información relacionada con la calidad del aire en el país, su ubicación y el tipo de equipos utilizados para su reporte.

El Subsistema de Información sobre Calidad del Aire es un sistema de información creado en cumplimiento de la Resolución 651 de 2010 para la captura, almacenamiento, transferencia, procesamiento y consulta de información. También permite la generación de información unificada de las redes de calidad del aire del país.

De acuerdo con el Artículo primero de la Resolución mencionada, el sistema debe constituirse en la fuente principal de información para el diseño, evaluación y ajuste de las políticas y estrategias nacionales y regionales de prevención y control de la contaminación del aire.

Formas de registrar información:

* Cargue manual en la página web de SISAIRE mediante el diligenciamiento de hojas de cálculo en formato Excel bajo una estructura predeterminada por el subsistema. Este procedimiento aplica principalmente para aquellos Sistemas de Vigilancia de la Calidad del Aire con equipos de medición de tipo manual.
* Cargue automático a la base de datos mediante Web Service. Este mecanismo aplica fundamentalmente para los Sistemas de Vigilancia de Calidad del Aire con equipos de medición de tipo automático.

## Propósito del servicio WEB de AIRE

Permitir a las autoridades ambientales que reporten al IDEAM información sobre calidad del aire hacerlo directamente desde sus aplicaciones utilizando estándares de interoperabilidad conocidos y confiables.

## ACERCA DE ESTA GUIA

El IDEAM dispone esta guía para facilitar el uso del anexo mencionado anteriormente, y está dirigida a personal técnico informático que se encargará el desarrollo de software para la integración del servicio web de SISAIRE con el sistema institucional de cada corporación.

IMPORTANTE: antes de de leer este documento verifique que cuenta con los Anexos que se describen en el documento “**WS\_RegistrarCalidadAire.doc**”, y que ha revisado sus contenidos.

# Responsabilidades Básicas del IDEAM y de la Autoridad Ambiental para adelantar la Integración

Es importante que tanto las corporaciones, entidades y autoridades ambientales realicen ciertas actividades administrativas y protocolarias para poder adelantar el proceso de integración informática con el IDEAM. De igual manera, el IDEAM es conciente de que debe desempeñar su rol de liderazgo para lograr este objetivo.

A continuación se enumeran las tareas más importantes para cada una de las partes que describen sus respectivos roles en el proceso.

**Autoridad ambiental:**

1. Manifestará al IDEAM su interés para iniciar el proceso de pruebas, indicando oportunamente su plan de trabajo y recursos asignados.
2. Preparará su información de acuerdo con los atributos mínimos requeridos por el sistema y las reglas de negocio.
3. Proveerá retroalimentación al IDEAM para ayudar a mejorar el servicio.

**IDEAM:**

1. Dispondrá un ambiente de pruebas y otro de producción en donde se expondrá el servicio web, de acuerdo con lo contemplado en la guía.
2. Anexará un plan de pruebas fijo o estándar para verificar el proceso de transmisión de información.
3. El acompañamiento del IDEAM consistirá en:
   1. En caso de algún error de consumo, ayudar a indentificar el punto de excepción, mediante la revisión del log del aplicativo en el servidor
   2. Disponer el acceso al log del aplicativo mediante un usuario de consulta a nivel de intranet
   3. Disponer el acceso a la base de datos del ambiente de pruebas, con un usuario de consulta

# Marco Técnico

## Estándares de Interoperabilidad

### Arquitectura Orientada al Servicio (SOA)

Para lograr que diferentes sistemas se comuniquen deben ser creados como componentes que cooperan mediante una implementación orientada a los servicios, y dichos servicios deben brindarse a través de interfaces de alto nivel que estén definidas sobre protocolos estandarizados.

El AIRE ha sido diseñado con una Arquitectura Orientada al Servicio (SOA):

* Desacoplando los diferentes niveles lógicos,
* Implementando y estableciendo en cada interfaz externa especificaciones como un modelo para el acceso a datos y métodos relacionados.
* Usando las interfaces que sirven tanto para el acceso interactivo via web como el programático desde aplicaciones externas.

Desde un punto de vista técnico, se eligió utilizar la tecnología SOAP de servicios web para facilitar la implementación y su consumo.

## Servicios WEB y el protocolo SOAP

Un servicio web es una interface que describe una colección de métodos, disponibles a través de la red y utilizando mensajes escritos en XML.

Gracias a los servicios web, una aplicación le permite a otras aplicaciones que interactúen con su métodos.

SOAP es un protocolo estándar, basado en XML, y sirve para acceder un servicio web.

Las especificaciones de SOAP siguen los estándares de W3C.

La solución via servicios web ofrece importantes ventajas arquitecturales:

* Permite reducir la dependencia entre las aplicaciones del cliente y del servidor (“acoplamiento débil”).
* Brinda un sistema de interoperabilidad independiente de la plataforma y las tecnologías.
* Permite la interoperabilidad a través de la red, ya que está basado en el protocolo SOAP que usualmente usa HTTP como protocolo básico (el cual es permitido en todo tipo de cortafuegos).
* Se basa en descriptores escritos en XML (WSDL).

# Ambiente de Pruebas de Consumo del Web Service

El IDEAM ha dispuesto un ambiente de pruebas que las autoridades ambientales pueden utilizar para realizar la implementación de sus clientes del servicio, sin afectar la información que se encuentra en el ambiente de producción. A continuación se brindan los datos de acceso a este ambiente.

### URLs del ambiente de pruebas

#### La URL de WSDL del ambiente de pruebas es:

<http://181.225.72.107:8080/Aire-SIAC-web-1.0/WS_IDEAM_RUA_RegistrarCalidadAire?wsdl>

# Consumo del Web Service

A continuación se presenta el Gráfico del proceso de consumo del WEB Service, en el cual un cliente una vez implementa la lógica del servicio, puede transferir información al Servidor:

## Descripción del anexo técnico de AIRE

El IDEAM ha creado la documentación de especificaciones técnicas del servicio web de AIRE para que sirva como herramienta de referencia en el momento de implementar aplicaciones que utilicen los métodos de AIRE para administrar y reportar información.

Esta documentación consiste en el documento de Word: WS\_RegistrarCalidadAire.doc

El documento tiene las siguientes secciones de interés práctico durante la implementación de cualquier aplicación cliente:

### Descripción del servicio, reglas del negocio, y detalle técnico de la base de datos

El anexo técnico tiene un capítulo que describe el servicio, sus reglas de negocio y detalles técnicos de la base de datos.

### Diccionario de datos

El diccionario de datos descrito en el anexo técnico le permitirá a la autoridad ambiental entender como se almacena finalmente la información que reporta al IDEAM.

### Especificacion de mensajes de requerimiento al servicio web

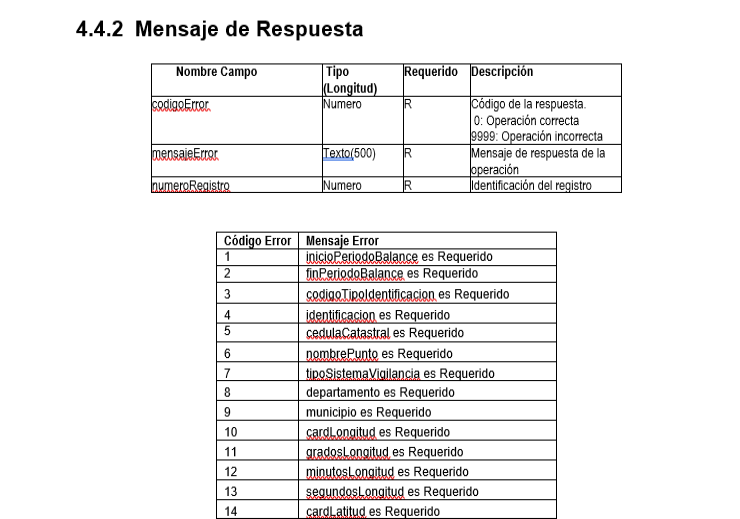
Se describen con detalle cada uno de los atributos de entrada a en los requerimientos de reporte de calidad de aire. Es aquí en donde la autoridad ambiental puede entender qué significa cada nombre de atributo, su tipo de datos y su obligatoriedad para lograr un reporte exitoso.

### Lista de autoridades ambientales

Con este listado la autoridad ambiental podrá averiguar cual es su código para enviar los reportes.

### Descripción de mensajes de respuesta

Como se puede ver en la siguiente imagen, en el capítulo 4.4.2 del anexo se describen los posibles códigos de éxito y error de respuesta por parte del servicio web:



## Preparación de la información en el Cliente

### Sincronización de las listas de valores

Para lograr que su aplicación se integre lo suficientemente con el AIRE será recomendable que mantenga una copia sincronizada de las tablas en las que se almacenan las listas de dominios y que su aplicación va a utilizar. Esto facilitará armár los llamados con los códigos y valores de lista correctos, así como la interpretación correcta de las respuestas del servicio web.

Para poder obtener el listado de valores controlados se puede utilizar la herramienta SoapUI.

Esta se puede configurar como se describe en el capítulo “DESCRIPCIÓN DE SOAPUI COMO HERRAMIENTA DE PRUEBAS DE LOS SERVICIOS WEB” de este documento. Debajo de cada ítem se anota el XML necesario para poder hacer el request de cada lista:

Las listas que se pueden sincronizar son las siguientes:

Valores de la tabla SIST\_LISTADO

Valores de la tabla SIST\_VALOR\_LISTADO

## Envío de Reportes

Como se mencionó anteriormente el servicio web de AIRE permitirá realizar los siguientes tipos de reportes:

* Registrar Calidad del Aire
* Registrar Emisiones
* Registrar Ruido

Cada uno de ellos requerirá que se utilicen los códigos de las listas de valores que el servicio web permite consumir.

En la versión actual del servicio web está habilitado el servicio de registro de calida del aire únicamente.

# Implementación del Cliente del Web Service

## Generación de Código Fuente del Cliente del WEB Service usando Netbeans

Netbeans tiene la funcionalidad de generar el código de Java necesario para poder consumir servicios WEB SOAP a partir de una URL (o archivo) de WSDL.

En los siguientes ejemplos se muestra como se puede consultar una lista de valores y como se puede realizar un reporte.

El consumo de una lista es importante, puesto que la autoridad ambiental tiene que conocer los códigos y nombres de las opciones de los parámetros que desea referir en su reporte. De lo contrario no será posible que el servicio web de SISAIRE pueda saber a qué entes está haciendo referencia. Todo se tiene que comunicar con códigos de IDEAM.

Para estos ejemplos se usarán proyectos web, pero la autoridad ambiental también puede elegir implementar el cliente del servicio web en una aplicación de tipo diferente, como, por ejemplo, una aplicación de escritorio.

Cabe anotar que los siguientes ejemplos tienen como propósito ilustrar cómo consumir el servicio web. No utilizan ningún patrón de diseño de software o framework especial. Por consiguiente, no se enfocarán en implementar rutinas de sincronización, conexión a bases de datos, uso de frameworks o patrones de diseño específicos. Será tarea de cada autoridad ambiental elegir estos elementos de acuerdo con las tecnologías y lineamientos que suele utilizar.

### Obtener Netbeans

##### Paso 1: Descargar Netbeans

Netbeans se puede descargar desde el siguiente enlace:

<https://netbeans.org/downloads/8.0.2/>

Para este ejemplo vamos a utilizar la versión 8.0.2 de Netbeans con soporte de Java EE.

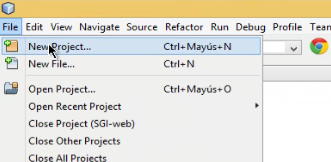
##### Paso 2: Instalar Netbeans

Para este paso, se recomienda seguir las instrucciones de instalación que vienen con Netbeans.

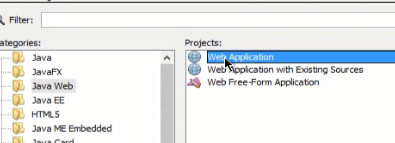
### Ejemplo: registrar la calidad del aire

##### Paso 1: Crear un nuevo proyecto de tipo WEB Service

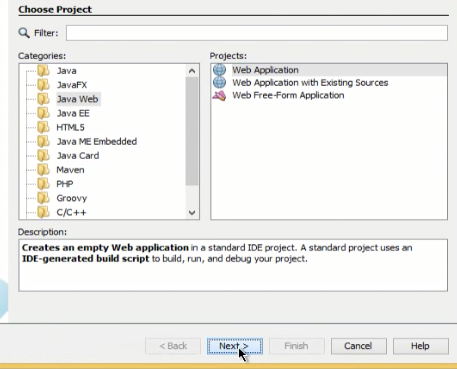
Vamos a File > New Project:



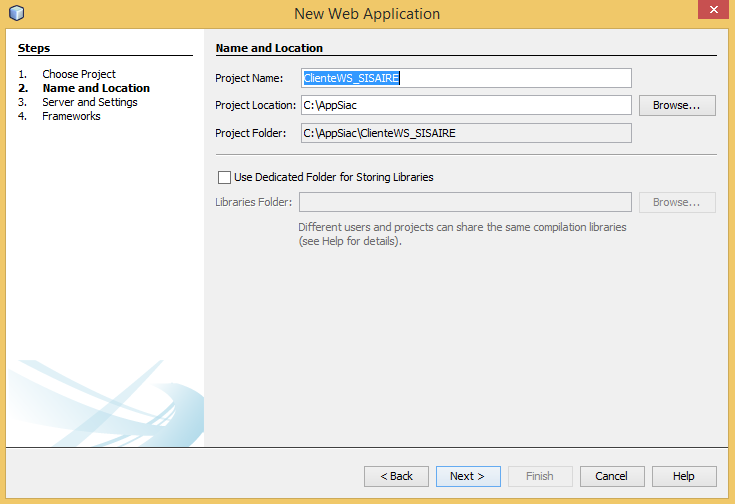
Seleccionamos Web Application:



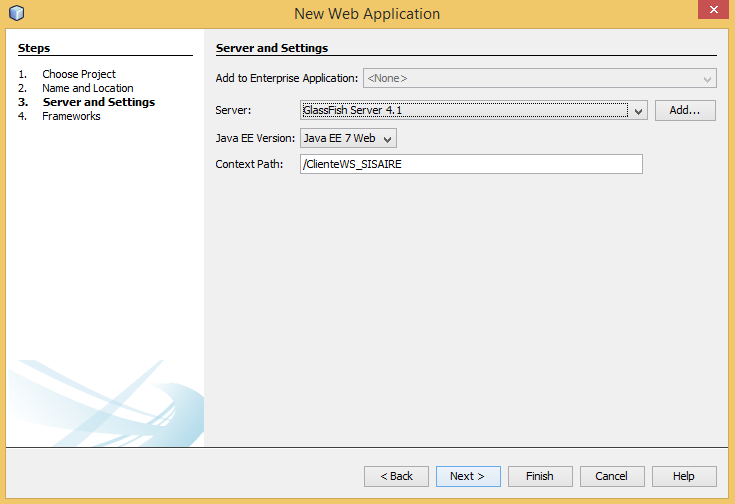
Clic en Next:



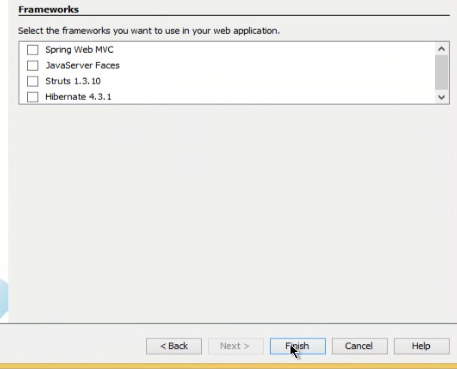
Especificamos el nombre dejamos la ubicación por defecto y clic en Next:



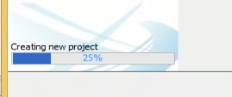
Dejamos el context path como está, click en next:



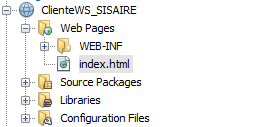
Clic en next nuevamente:



Esperamos a que termine de crear el nuevo proyecto:

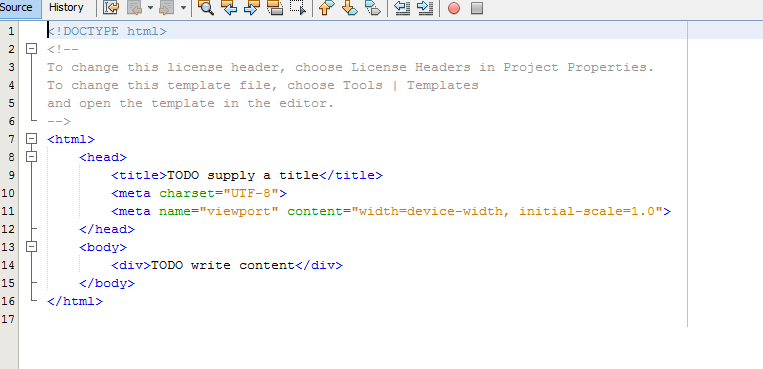


Podemos ver que creo una página index.html por defecto. Modificaremos y usaremos esa página para acceder a la opción de sincronización de estaciones:

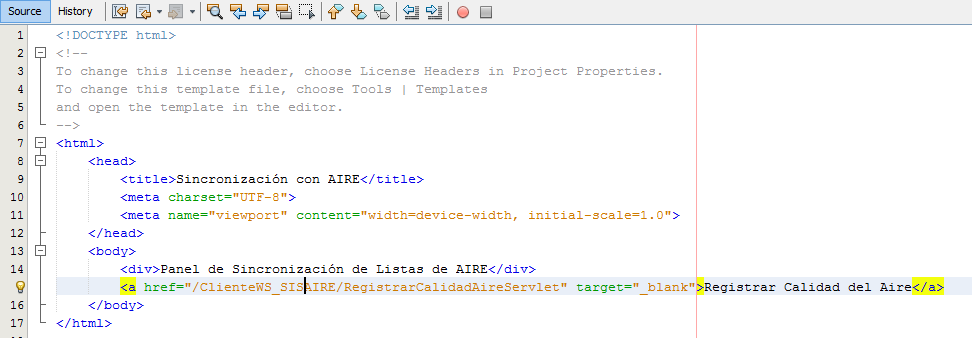


#### Paso 2: agregar la opción de registrar la calidad del aire a la página de inicio:

Antes de la edición:

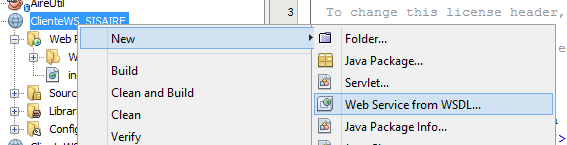


Después de la edición:

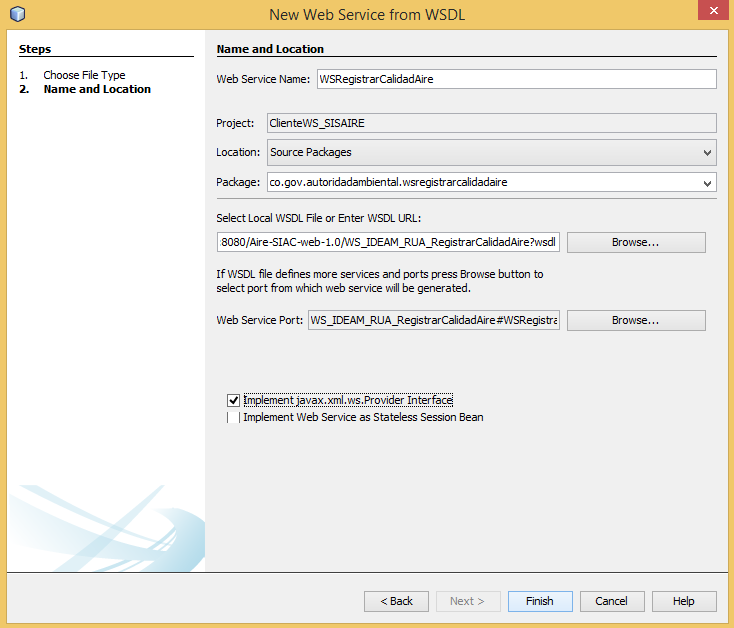


#### Paso 3: generar las clases de registro de calidad de aire a partir del WSDL respectivo

Creamos el cliente del servicio web:

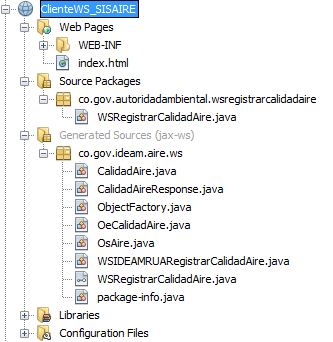


Definimos el nombre y ubicación del cliente del web service, el cual pondremos en un nuevo paquete:



Importante: es necesario obtener la URL del WSDL de la Oficina de Infomática del IDEAM, y además tener acceso VPN para poder acceder a la IP de donde se sirve el web service. Para esto se debe contactar a la Oficina Informática del IDEAM.

Como resultado se habrán creado el paquete y las clases correspondientes para poder consumir el servicio web:

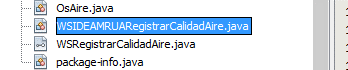


#### Paso 4: implementar la lógica del consumo

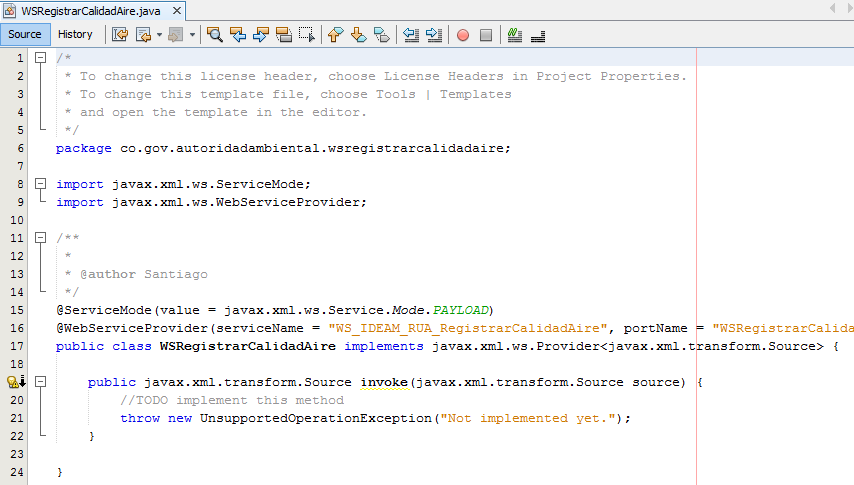
Netbeans genera automáticamente las clases de Java para poder consumir el servicio web de acuerdo con el WSDL que consultó en la URL de WSDL. Dicho código está listo para ser usado por las funciones de la aplicación de la autoridad ambiental.

###### Explicación del código generado

Las clases proxy son para este servicio web ahora WSIDEAMRUARegistrarCalidadAire.java, y WSRegistrarCalidadAire.java:



Y la clase por defecto generada para consumir el servicio es WSRegistrarCalidadAire.java:



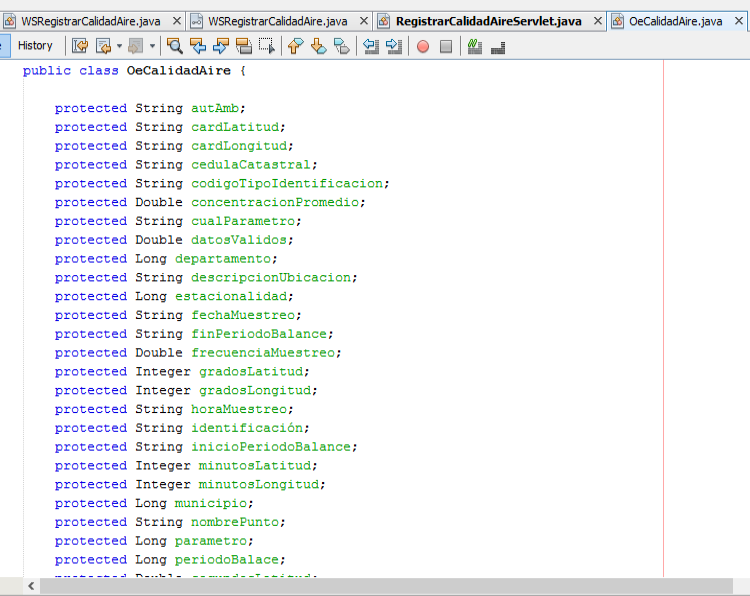
**¿Cómo enviar la información?**

Al visualizar la clase WSRegistrarCalidadAire.java, la cual declara el método para enviar un reporte de calidad de aire podemos ver qué parámetros requiere:



Vemos que el parámetro que espera es de tipo OeCalidadAire.

Ahora, para poder saber qué atributos se deben definir al instanciar el objeto OeCalidadAire, nos remitimos a la clase OeCalidadAire.java:

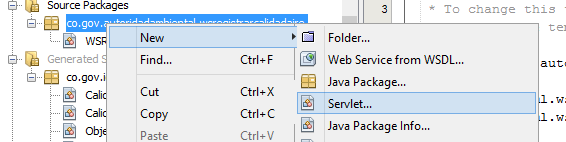


De este modo, en nuestro método processRequest del servlet que creamos y que estamos usando para ejecutar el registro de calidad de aire, solo tenemos que instanciar la clase de OeCalidadAire y definir sus atributos. Luego se hace el llamado al método CalidadAire del servicio enviando como parámetro dicho objeto.

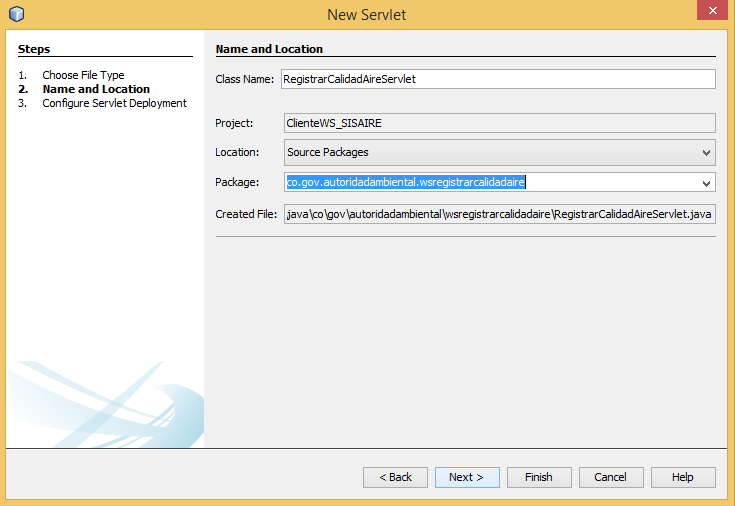
WSRegistrarCalidadAire.java es una clase que nos puede servir para implementar los métodos de consulta al servicio web. No obstante, más adelante vamos a crear nuestro propio servlet SincronizarEspecies.java para consultar la información.

#### Paso 5: crear una clase para de consumir el método CalidadAire de WS\_IDEAM\_RUA\_RegistrarCalidadAire

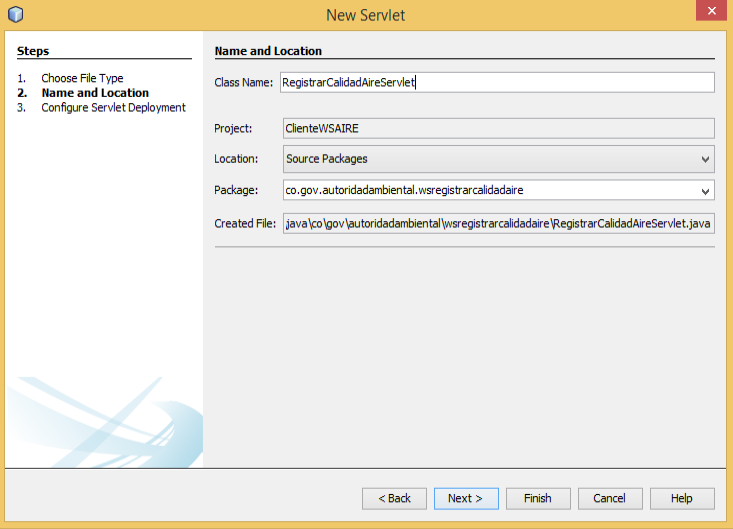
En nuestro caso, vamos a crear un servlet nuevo llamado RegistrarCalidadAire.java:



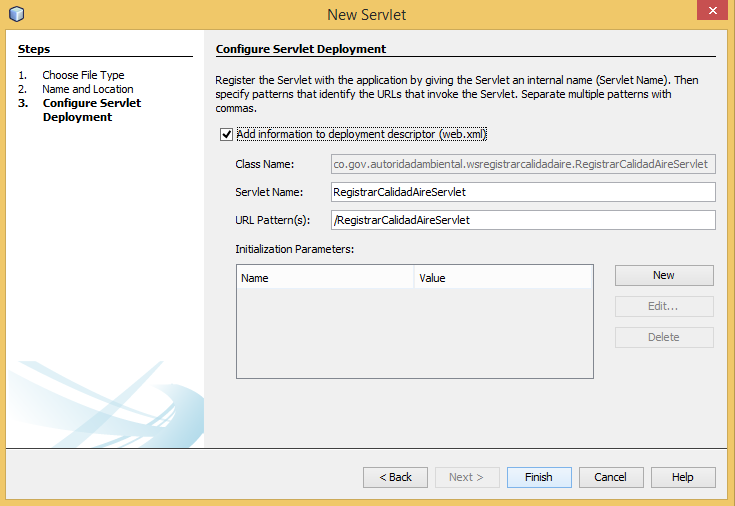
Especificamos el nombre y la ubicación:



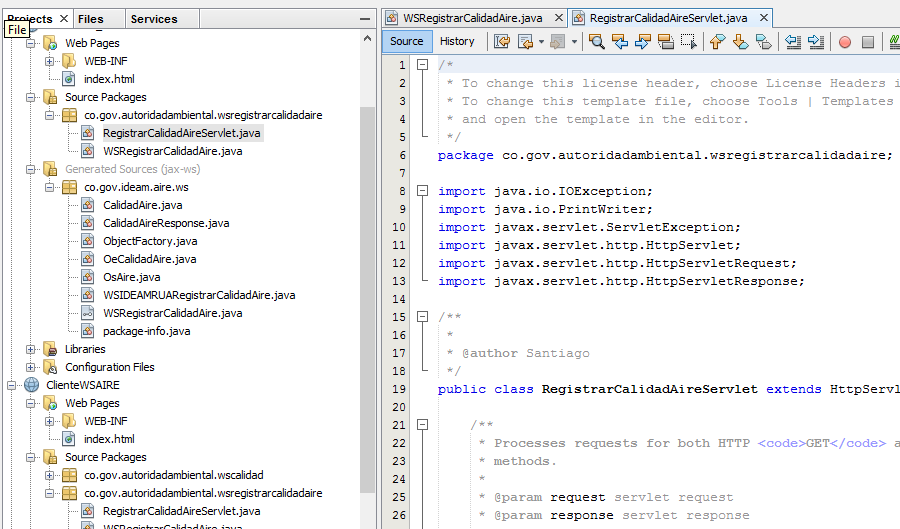
Configuramos el despliegue del servlet agregándolo al descriptor:



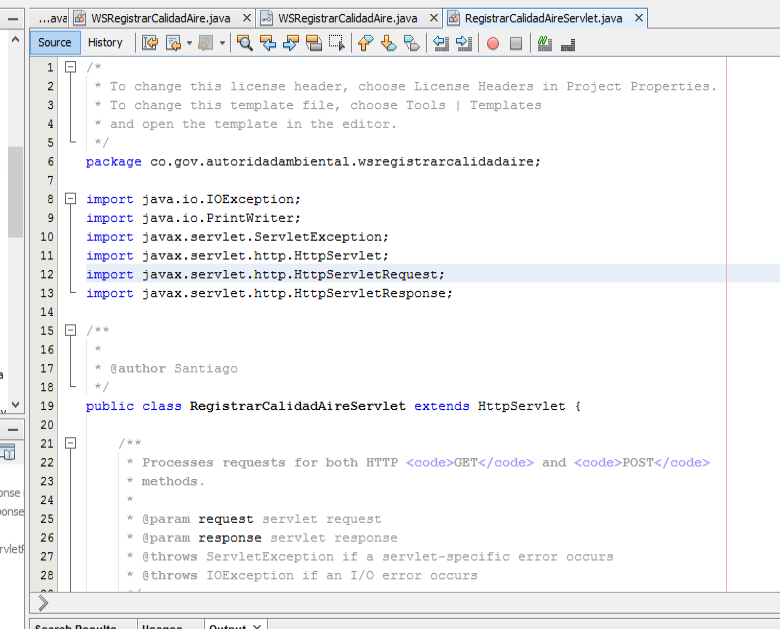
Configuramos el despliegue del Servlet:



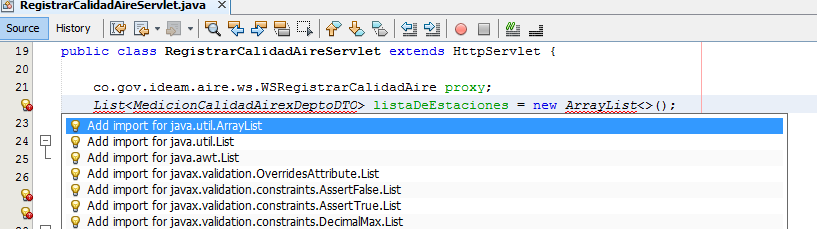
Debe quedar creado dentro del paquete:



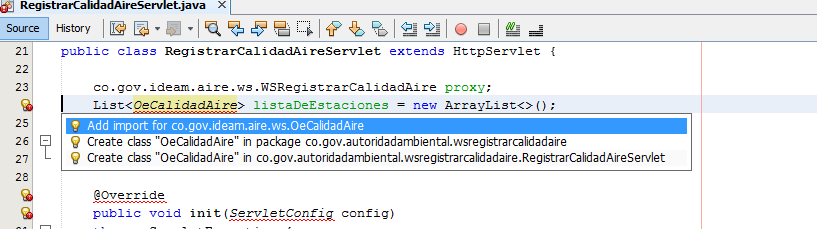
El servlet se debería ver así, antes de editarlo:



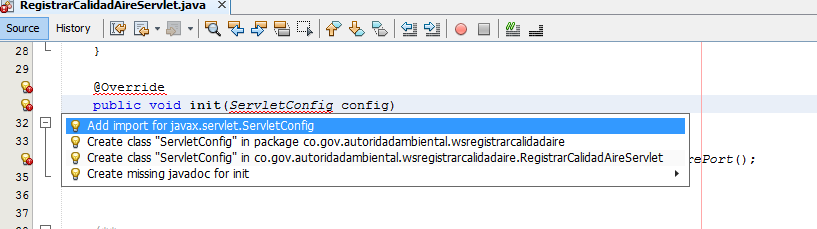
Comenzamos a ingresar las siguientes líneas y agregamos los imports que falten (clic sobre el bombillo):



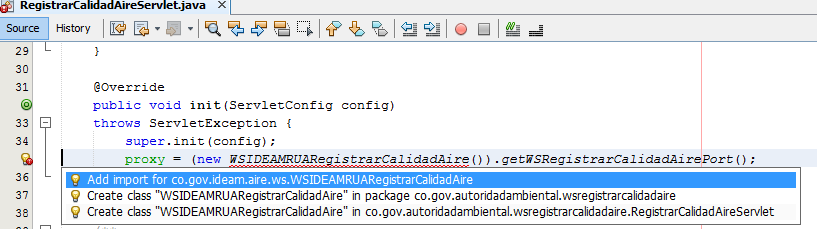
Importamos también la clase OeCalidadAire.java, que sirve para transportar todo el registro de calidad de aire, dando clic sobre el bombillo:



Importamos ServletConfig:

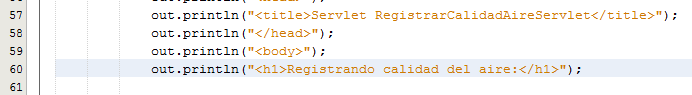


Importamos WSIDEAMRUARegistrarCalidadAire para poder acceder los métodos del servicio:



Luego, agregamos el código para consumir el método y desplegar el resultado. Es importante mencionar que, por el propósito y alcance de este documento, en este ejemplo la información de que se enviará al servicio web se especifica manualmente en el código. Cuando la autoridad ambiental implemente este cliente, deberá obtener la información desde su base de datos.

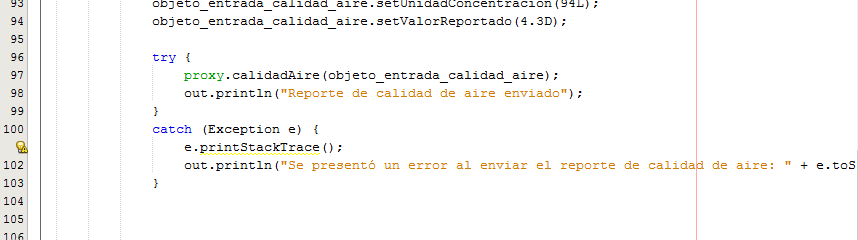
Cambiamos los mensajes del servlet:



Instanciamos el objeto de entrada OeCalidadAire que nos sirve para transportar el registro del reporte de calidad y especificamos los parámetros (más adelante se muestra el código):



Y hacemos el llamado al método para enviar el reporte:



El código final del servlet es:

package co.gov.autoridadambiental.wsregistrarcalidadaire;

import co.gov.ideam.aire.ws.OeCalidadAire;

import co.gov.ideam.aire.ws.WSIDEAMRUARegistrarCalidadAire;

import java.io.IOException;

import java.io.PrintWriter;

import java.util.ArrayList;

import java.util.List;

import javax.servlet.ServletConfig;

import javax.servlet.ServletException;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

/\*\*

\*

\* @author Santiago

\*/

public class RegistrarCalidadAireServlet extends HttpServlet {

co.gov.ideam.aire.ws.WSRegistrarCalidadAire proxy;

List<OeCalidadAire> listaDeEstaciones = new ArrayList<>();

public RegistrarCalidadAireServlet() {

}

@Override

public void init(ServletConfig config)

throws ServletException {

super.init(config);

proxy = (new WSIDEAMRUARegistrarCalidadAire()).getWSRegistrarCalidadAirePort();

}

/\*\*

\* Processes requests for both HTTP <code>GET</code> and <code>POST</code>

\* methods.

\*

\* @param request servlet request

\* @param response servlet response

\* @throws ServletException if a servlet-specific error occurs

\* @throws IOException if an I/O error occurs

\*/

protected void processRequest(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

response.setContentType("text/html;charset=UTF-8");

try (PrintWriter out = response.getWriter()) {

/\* TODO output your page here. You may use following sample code. \*/

out.println("<!DOCTYPE html>");

out.println("<html>");

out.println("<head>");

out.println("<title>Servlet RegistrarCalidadAireServlet</title>");

out.println("</head>");

out.println("<body>");

out.println("<h1>Registrando calidad del aire:</h1>");

OeCalidadAire objeto\_entrada\_calidad\_aire = new OeCalidadAire();

objeto\_entrada\_calidad\_aire.setAutAmb("ANLA");

objeto\_entrada\_calidad\_aire.setCardLatitud("NORTE");

objeto\_entrada\_calidad\_aire.setCardLatitud("OESTE");

objeto\_entrada\_calidad\_aire.setCedulaCatastral("OF");

objeto\_entrada\_calidad\_aire.setCodigoTipoIdentificacion("NIT");

objeto\_entrada\_calidad\_aire.setConcentracionPromedio(4.56);

objeto\_entrada\_calidad\_aire.setCualParametro("");

objeto\_entrada\_calidad\_aire.setDatosValidos(99D);

objeto\_entrada\_calidad\_aire.setDepartamento(0L);

objeto\_entrada\_calidad\_aire.setDescripcionUbicacion("Prueba de Ubicación");

objeto\_entrada\_calidad\_aire.setEstacionalidad(77L);

objeto\_entrada\_calidad\_aire.setFechaMuestreo("01/03/2016");

objeto\_entrada\_calidad\_aire.setFinPeriodoBalance("31/12/2015");

objeto\_entrada\_calidad\_aire.setFrecuenciaMuestreo(8D);

objeto\_entrada\_calidad\_aire.setGradosLatitud(6);

objeto\_entrada\_calidad\_aire.setGradosLongitud(73);

objeto\_entrada\_calidad\_aire.setHoraMuestreo("10:13:00 AM");

objeto\_entrada\_calidad\_aire.setIdentificación("3102851087");

objeto\_entrada\_calidad\_aire.setInicioPeriodoBalance("01/01/2015");

objeto\_entrada\_calidad\_aire.setMinutosLatitud(5);

objeto\_entrada\_calidad\_aire.setMinutosLongitud(33);

objeto\_entrada\_calidad\_aire.setMunicipio(907L);

objeto\_entrada\_calidad\_aire.setNombrePunto("LA COCHA ");

objeto\_entrada\_calidad\_aire.setParametro(377L);

objeto\_entrada\_calidad\_aire.setPeriodoBalace(5000092005L);

objeto\_entrada\_calidad\_aire.setSegundosLatitud(44D);

objeto\_entrada\_calidad\_aire.setSegundosLongitud(43D);

objeto\_entrada\_calidad\_aire.setTiempoMuestraDias(4D);

objeto\_entrada\_calidad\_aire.setTiempoMuestraHoras(24D);

objeto\_entrada\_calidad\_aire.setTipoSistemaVigilancia(4L);

objeto\_entrada\_calidad\_aire.setUnidadConcentracion(94L);

objeto\_entrada\_calidad\_aire.setValorReportado(4.3D);

try {

proxy.calidadAire(objeto\_entrada\_calidad\_aire);

out.println("Reporte de calidad de aire enviado");

}

catch (Exception e) {

e.printStackTrace();

out.println("Se presentó un error al enviar el reporte de calidad de aire: " + e.toString());

}

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

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

}

}

// <editor-fold defaultstate="collapsed" desc="HttpServlet methods. Click on the + sign on the left to edit the code.">

/\*\*

\* Handles the HTTP <code>GET</code> method.

\*

\* @param request servlet request

\* @param response servlet response

\* @throws ServletException if a servlet-specific error occurs

\* @throws IOException if an I/O error occurs

\*/

@Override

protected void doGet(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

/\*\*

\* Handles the HTTP <code>POST</code> method.

\*

\* @param request servlet request

\* @param response servlet response

\* @throws ServletException if a servlet-specific error occurs

\* @throws IOException if an I/O error occurs

\*/

@Override

protected void doPost(HttpServletRequest request, HttpServletResponse response)

throws ServletException, IOException {

processRequest(request, response);

}

/\*\*

\* Returns a short description of the servlet.

\*

\* @return a String containing servlet description

\*/

@Override

public String getServletInfo() {

return "Short description";

}// </editor-fold>

}

De este código es de resaltar varias cosas:

En al gunos casos fue necesario poner una L, o D en los parámetros de definición de los atributos par indicarle al complador de que se trata de un entero largo o un doble.

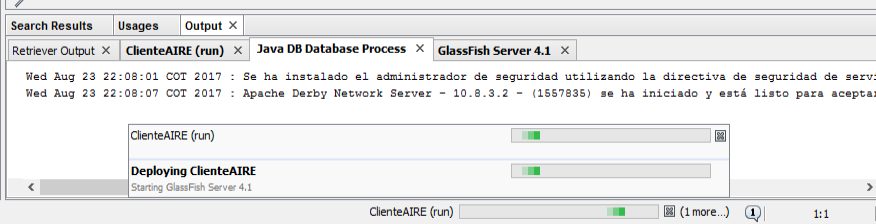
Para saber qué datos y valores especificar para cada parámetro, se consulta el anexo técnico que especifica los valores válidos para cada variable de este método o, alternativamente la clase OeCalidadAire.java.

##### Paso 4: lanzar aplicación web

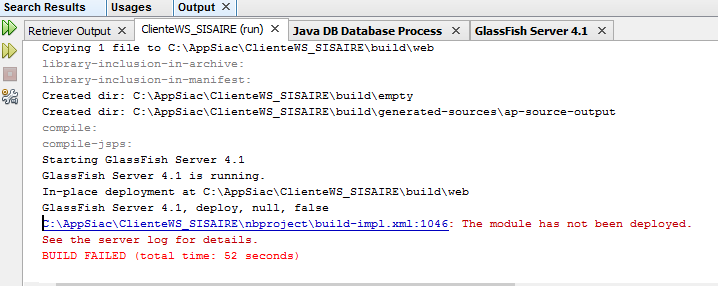
Netbeans viene por defecto con el servicio de GlassFish Server que nos permite desplegar y visualizar nuestra aplicación web. Será decisión de cada autoridad ambiental elegir y configurar el servidor de aplicaciones que prefiera. Solo tenemos que dar clic en el botón de iniciar que se encuentra en la parte superior:



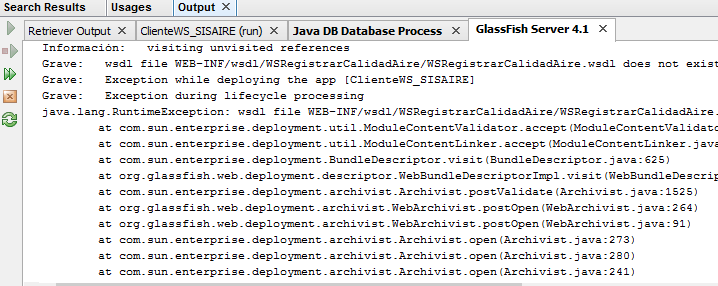
Esperamos a que el servicio de glassfish acabe de iniciar.



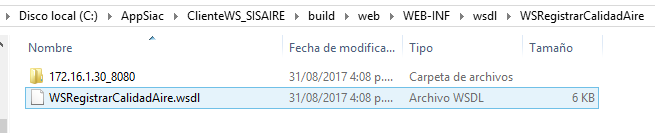
Si sale un error diciendo que la aplicación ClienteWSAire no se pudo desplegar (deploy):



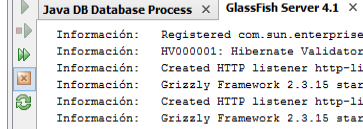
y pone como causa que el archivo WSCalidad.wsdl no se encuentra en cierta ruta:



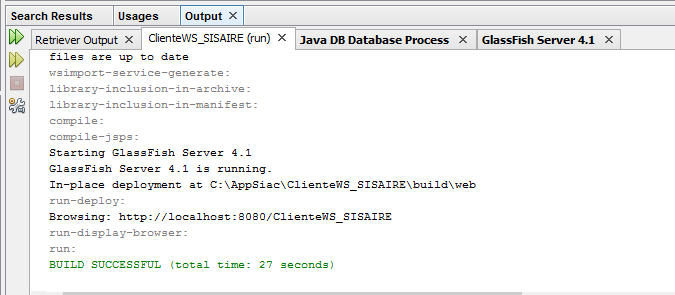
entonces procedemos a copiar el archivo WS\_IDEAM\_RUA\_RegistrarCalidadAire.wsdl desde la ruta (reemplazar c:\AppSiac por la ruta que haya elegido en su computador para ubicar la aplicación) C:\AppSiac\ClienteWS\_SISAIRE\build\web\WEB-INF\wsdl\WSRegistrarCalidadAire\ 172.16.1.30\_8080\Aire-SIAC-web-1.0 a la ruta C:\AppSiac\ClienteWS\_SISAIRE\build\web\WEB-INF\wsdl\WSCalidad y lo renombramos a WSRegistrarCalidadAire.wsdl:



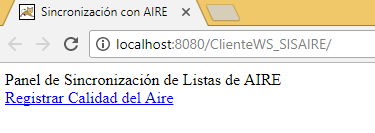
Detenemos el servicio de GlassFish dando clic en el botón cuadrado rojo:



Volvemos a dar clic en el botón verde para ejecutar el proyecto. Debería aparecer BUILD SUCCESFUL:

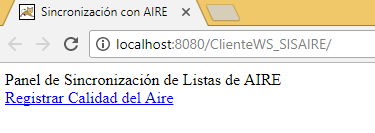


Si no se abre la página automáticamente, probamos que la aplicación esté ejecutándose abriendo la página <http://localhost:8080/ClienteWSAIRE/> :



#### Paso 6: Probar el consumo del servicio de registro de calidad de aire

Continuamos desde el paso anterior:



Damos clic en Registrar Calidad de Aire y recibimos confirmación de que la operación fue exitosa:



Después de esto se puede pasar a la fase de verificación. El hecho de que no haya generado una excepción y salido el mensaje de error que definimos en el código es un buen indicio de que el registro si fue enviado.

## Descripción de SoapUI como herramienta de pruebas de los servicios web

### Descarga de SoapUI

Se puede descargar SopaUI desde el siguiente enlace:

<https://www.soapui.org/downloads/soapui.html>

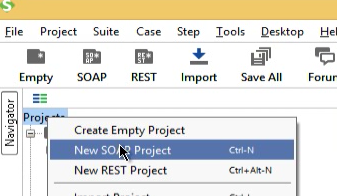
Se elige la opción de descargar SoapUI OpenSource.

### Instalación de SoapUI

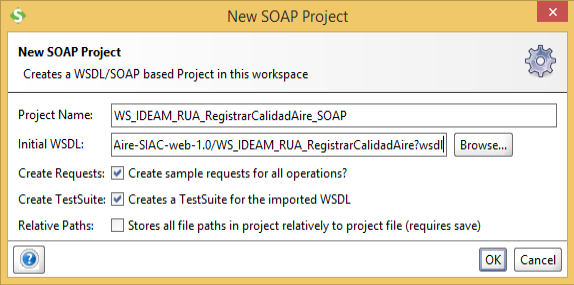
Para instalar SoapUI se siguen las instrucciones del proveedor. Estas se encuentran en la página:

<https://www.soapui.org/getting-started/installing-soapui.html>

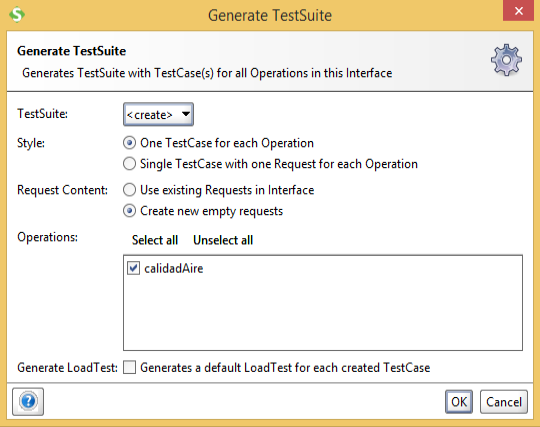
### Creación de un proyecto en SoapUI

Para crear un proyecto damos clic derecho sobre Projects y elegimos la opción New SOAP Project:  


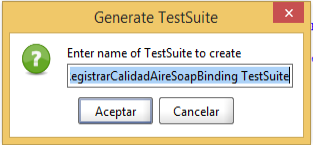
Especicamos el nombre, la URL de WSDL del ambiente deseado y demás opciones que aparecen en la imagen:



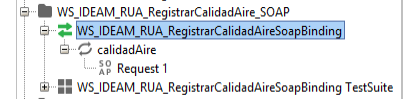
Esp eramos que termine de procesar el WSDL. Cuando pregunte por los detalles de generación de pruebas especificamos según la siguiente imagen:



Luego especificamos el nombre del conjunto de pruebas:

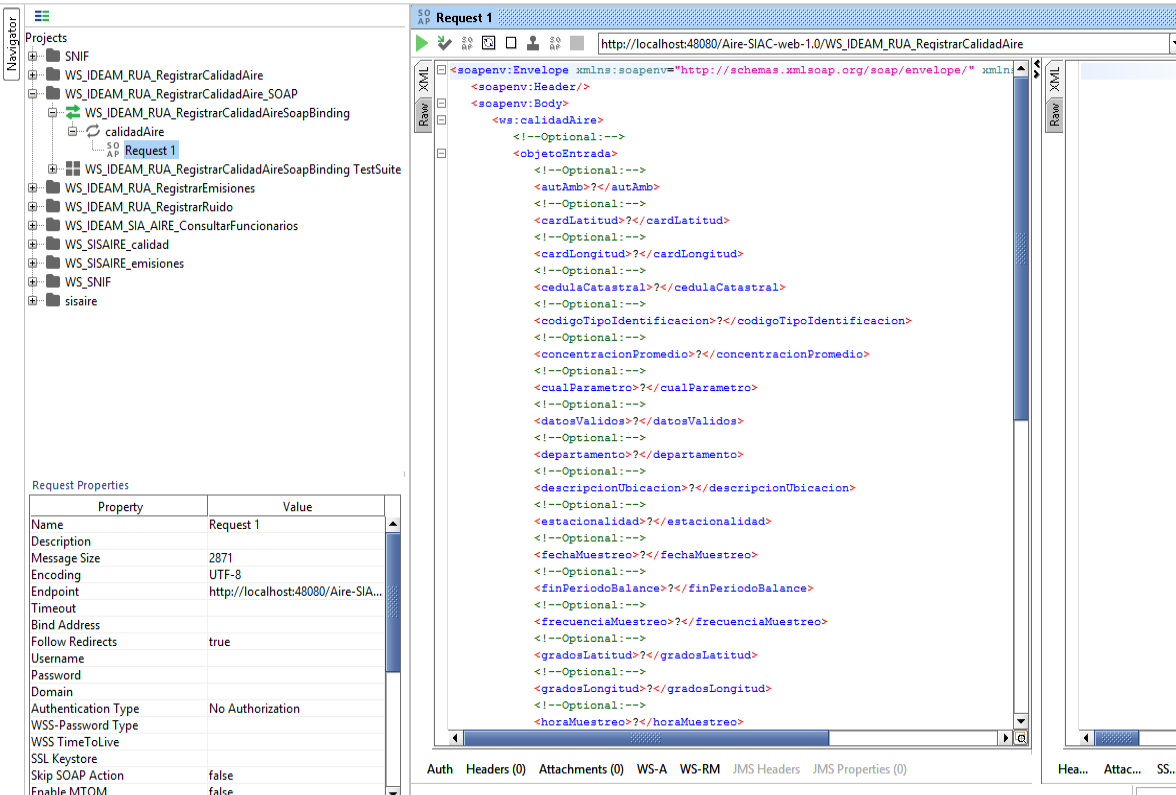


Una vez finalizado el proceso de creación del proyecto aparecerá en el listado de proyectos:



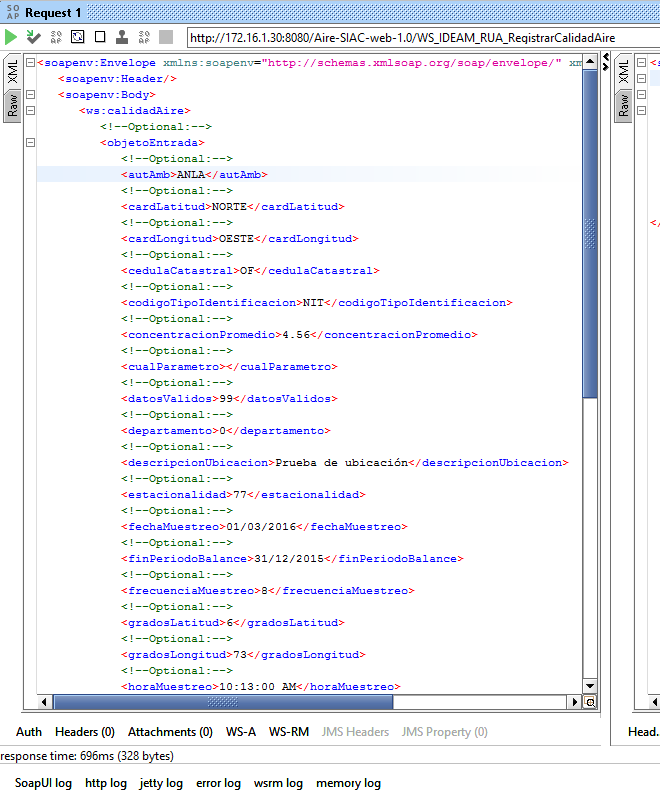
### Prueba del servicio WEB

Procedemos a abrir con doble clic el requerimiento de prueba Request 1 del método calidadAire:

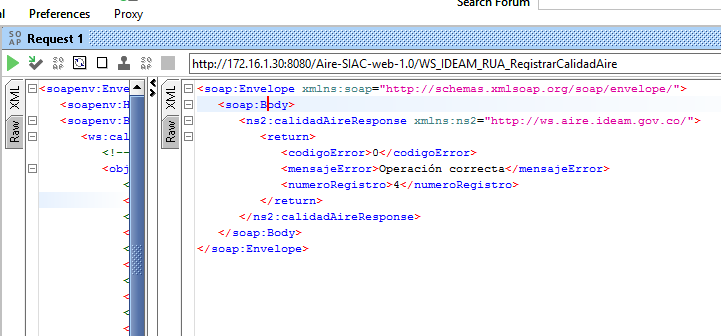


Vemos que, por defecto, los parámetros del método tienen signos de interrogación como valor. Estos deben ser reemplazados por los valores correctos. Comenzamos entonces, refiriéndonos al anexo técnico de especificaciones del web service a diligenciar manualmente estos valores.

Finalmente tendremos un XML similar al siguiente:



Damos clic en el botón de ejecutar y esperamos a que se listen todas las especies en la respuesta XML del servicio web:



### Interpretación del Mensaje de Respuesta

La forma en la que el servicio web comunica el resultado del consumo se puede consultar en el anexo. Cada método cuenta con una sección que explica qué retorna.

El mensaje nos permite entender si la operación de consumo tuvo éxito, o, por el contrario se presentó algún error. Será útil, especialmente si se presenta algún error poder entender cual es su causa y tomar medidas al respecto. En el anexo de especificaciones se encuentra, para cada método, una tabla que enumera los posibles mensajes de error u excepción.

### Verificación de la Información Reportada

Se deberá contactar al IDEAM para poder obtener un listado de registros reportados.