

Introducción al desarrollo de aplicaciones en red

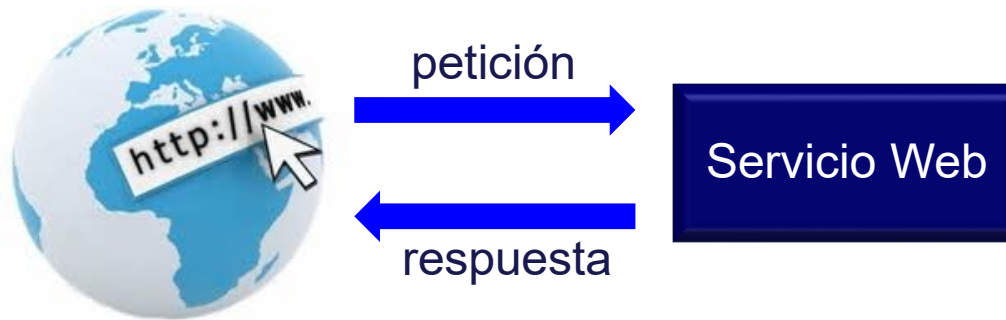
Programación de redes, sistemas y servicios

- Introducción a los servicios Web
 - Servicios web *SOAP*
 - Servicios web *RESTful*
- Introducción a los servicios Web en Java
 - JAX-WS y JAX-RS
 - Ejemplos de implementación de clientes de servicios Web
 - Ejercicios de implementación de servicios web

Introducción a los servicios Web

Introducción

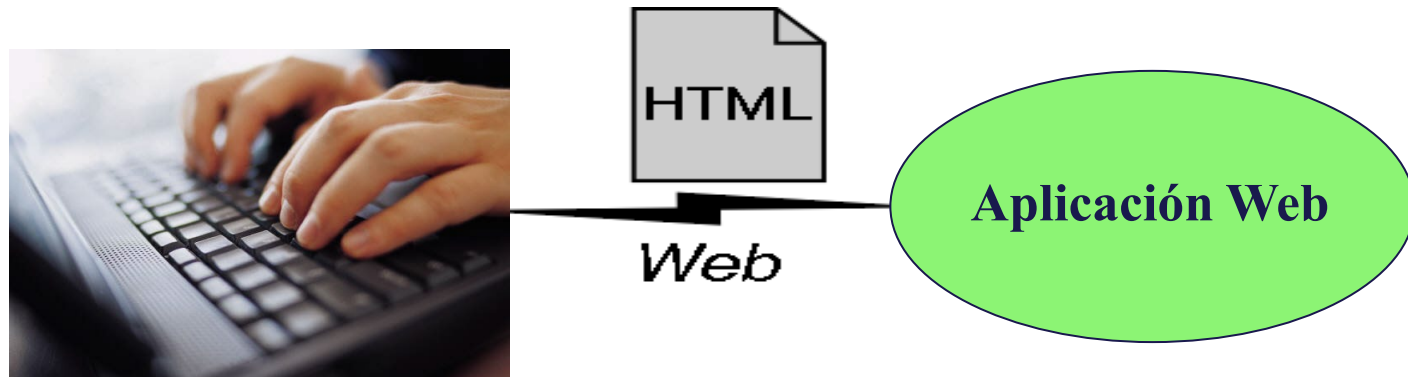
- Un *servicio Web* es una aplicación cuya funcionalidad es accesible a través del protocolo *http*, es decir, puede ser publicado, localizado e invocado a través de la Web



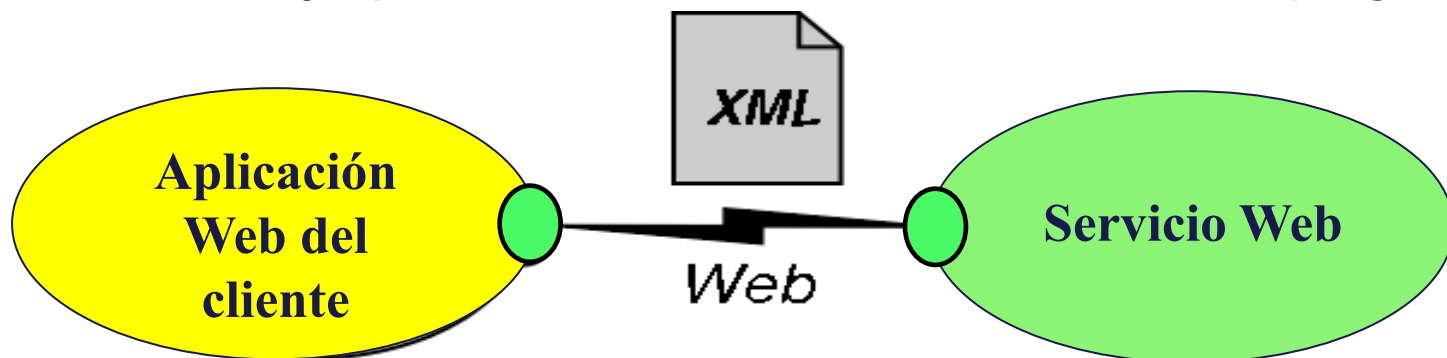
- Los clientes envían solicitudes de ejecución de métodos con sus argumentos
- Los servidores responden enviando los resultados
- Facilita la comunicación entre aplicaciones en entornos heterogéneos (independiente del lenguaje y de la plataforma o sistema operativo)

Introducción a los servicios Web

Aplicación Web tradicional vs Servicio Web



Una *aplicación Web* es cualquier aplicación que reside en un servidor y que es accedida a través de una *página Web*



El *servicio Web* es similar al anterior, pero está orientada al uso desde otras aplicaciones

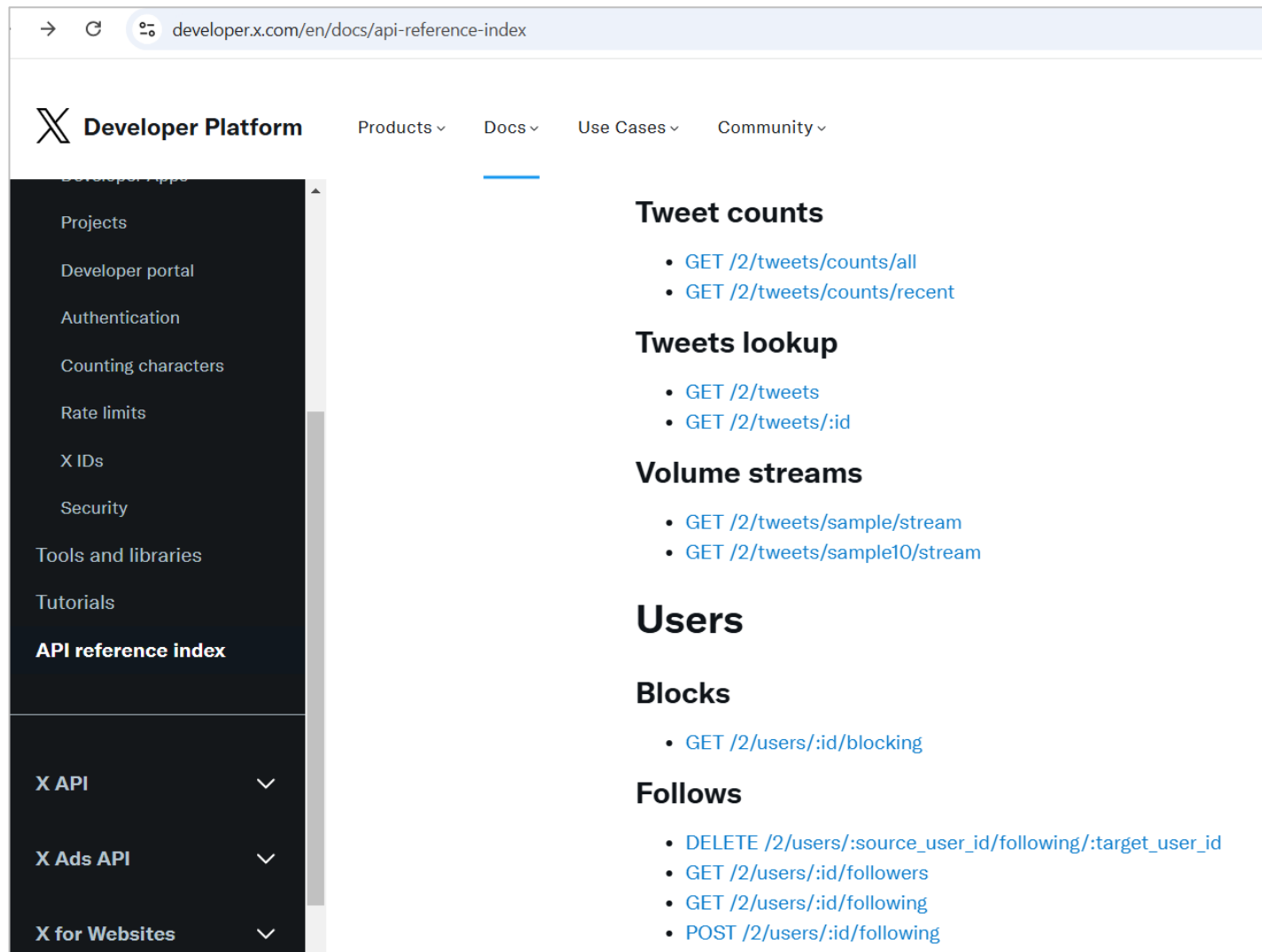
Introducción a los servicios Web

Implementación de los servicios Web

- Los **servicios Web** se puede implementar mediante:
 - **Servicios Web SOAP**,
 - Las peticiones del cliente y las respuestas del servidor son mensajes escritos en XML (*mensajes SOAP*), que se incluyen en el cuerpo de un paquete HTTP
 - El mensaje SOAP del cliente incluye la operación deseada y los argumentos necesarios. El mensaje SOAP del servidor incluye el resultado de la operación solicitada
 - Las operaciones a realizar son los métodos asociados al servicio
 - **Servicios Web RESTful**,
 - Se basa en elementos de información (*recursos*), identificados mediante una URI
 - Para manipular los recursos se usan las operaciones propias de HTTP (POST, GET, PUT y DELETE), junto con los argumentos necesarios
 - El formato de los datos intercambiados vía HTTP puede ser texto (sin formato), HTML, XML, JSON, etc.

Introducción a los servicios Web

Implementación de los servicios Web



The screenshot shows the Twitter Developer Platform API reference index page. The browser address bar displays `developer.x.com/en/docs/api-reference-index`. The page features a dark sidebar on the left with a list of navigation items: Projects, Developer portal, Authentication, Counting characters, Rate limits, X IDs, Security, Tools and libraries, Tutorials, API reference index (highlighted), X API, X Ads API, and X for Websites. The main content area is titled "Developer Platform" and includes navigation links for Products, Docs (active), Use Cases, and Community. The "API reference index" section lists several API endpoints under different categories:

- Tweet counts**
 - [GET /2/tweets/counts/all](#)
 - [GET /2/tweets/counts/recent](#)
- Tweets lookup**
 - [GET /2/tweets](#)
 - [GET /2/tweets/:id](#)
- Volume streams**
 - [GET /2/tweets/sample/stream](#)
 - [GET /2/tweets/sample10/stream](#)
- Users**
- Blocks**
 - [GET /2/users/:id/blocking](#)
- Follows**
 - [DELETE /2/users/:source_user_id/following/:target_user_id](#)
 - [GET /2/users/:id/followers](#)
 - [GET /2/users/:id/following](#)
 - [POST /2/users/:id/following](#)

Introducción a los servicios Web

Implementación de los servicios Web

- SOAP

- Más complejo
- El intercambio se realiza sólo mediante mensajes XML
- Más seguro (SSL y permite integrar otros protocolos que aportan mayor seguridad en las comunicaciones)
- Permite la recuperación ante fallos de la comunicación

SOAP puede entenderse como el envío de solicitudes y respuestas mediante sobres de correos

- REST

- Menos complejo (menos formalidades)
- Más rápido
- Permite mayor variedad de formatos de datos
- Menos seguro (SSL/TLS)
- Ante un fallo de comunicación hay que empezar de nuevo (sin estado)

REST puede verse como el envío de solicitudes y respuestas mediante postales

Introducción a los servicios Web

Implementación de los servicios Web

- El *lenguaje de marcado extensible* (XML) es un lenguaje de marcado que permite definir y almacenar datos, facilitando la transmisión de datos a través de cualquier red
- Utiliza *etiquetas* junto con unas *reglas predefinidas* para definir los datos
- Por ejemplo, para definir datos de una librería se puede usar etiquetas como <libro>, <título> y <autor>, resultando en:
 <libro>
 <título>Cien años de soledad</título>
 <autor>Gabriel García Márquez</autor>
 </libro>

Introducción a los servicios Web

Mensajes SOAP

**Implementación
servicio Web**



mensaje SOAP

solicitud

POST /StockPrice HTTP/1.1

Host: www.example.org

Content-Type: text/xml; charset=utf-8

Content-Length: nnn

HTTP (petición SOAP)

<?xml version="1.0"?>

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">

<soap:Header>

<!-- Encabezado opcional (detalles seguridad, transacciones, etc) -->

</soap:Header>

<soap:Body>

<!-- Datos de la solicitud -->

</soap:Body>

</soap:Envelope>

respuesta



Cliente

HTTP/1.1 200 OK

Content-Type: text/xml; charset=utf-8

Content-Length: mmm

HTTP (respuesta SOAP)

<?xml version="1.0"?>

<soap:Envelope xmlns:soap="http://www.w3.org/2003/05/soap-envelope">

<soap:Header>

<!-- Encabezado opcional -->

</soap:Header>

<soap:Body>

<!-- Respuesta -->

</soap:Body>

</soap:Envelope>

mensaje SOAP

Introducción a los servicios Web

Arquitectura orientada a servicios: WSDL

- **WSDL** (*Web Services Description Languages*) es un documento XML que describe los métodos que están disponibles en un *servicio web* **SOAP** (operaciones, parámetros y tipos de datos)
- El documento **WSDL** está accesible en el *servidor* que ofrece el servicio o en un directorio de servicios
- Consultando el documento **WSDL** los clientes conocen la forma de interaccionar con cada servicio:
 - ¿qué operaciones están disponibles?
 - ¿qué parámetros espera recibir (tipos)?
 - ¿devuelve algún resultado (tipo de retorno)? ...

Introducción a los servicios Web

Servicios Web RESTful

- **REST** está orientado a recursos (en vez de a servicios)
 - Los recursos son los objetos, datos, etc. a los que puede acceder un cliente
 - Los recursos se identifican de forma única mediante **URIs**
- Sobre los recursos se puede realizar operaciones como LEER, CREAR, ACTUALIZAR y BORRAR, mediante los métodos HTTP: **GET**, **POST**, **PUT** y **DELETE**
- Los recursos se pueden representar en varios formatos: XML, JSON, PNG, etc.
 - Cada operación debe definir los tipos de formato soportados

Introducción a los servicios Web

Servicios Web RESTful

GET /api/v1/network-device HTTP/1.1

Host: api.codnova.com

Accept: */*

HTTP (petición REST)

Implementación servicio Web



solicitud

respuesta



Cliente

HTTP/1.1 200 OK

Content-Type: application/json; charset=utf-8

HTTP (respuesta REST)

```
{
  "Routers": [
    {
      "Hostname": "R1"
      "Vendor": "Cisco"
      "Id": "1"
    },
    {
      "Hostname": "R2"
      "Vendor": "Huawei"
      "Id": "2"
    },
  ]
}
```

Formato JSON

- Consiste en pares "clave": valor
- Mediante llaves {} se definen objetos
- Mediante corchetes [] se definen arrays

- Introducción a los servicios Web
 - Servicios web *SOAP*
 - Servicios web *RESTful*
- Introducción a los servicios Web en Java
 - JAX-WS y JAX-RS
 - Ejemplos de implementación de clientes de servicios Web
 - Ejercicios de implementación de servicios web

Introducción a los servicios Web en Java

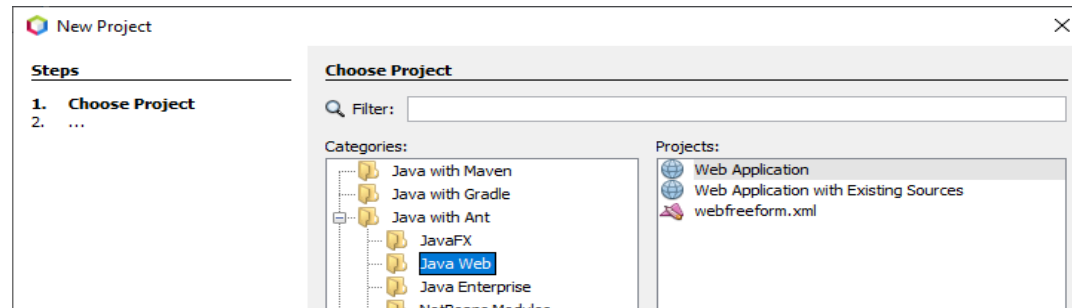
Introducción

- Existen diversas tecnologías usadas en el desarrollo de *servicios Web en Java* como *JAX-WS* (para SOAP) y *JAX-RS* (para REST)
- Existen frameworks que implementan estas tecnologías como *Axis2*, *Metro*, *Jersey*, etc.
- Para implementar un *servicio Web* se necesita un *servidor de aplicaciones* que incluya alguno de estos frameworks, como Glassfish o Payara

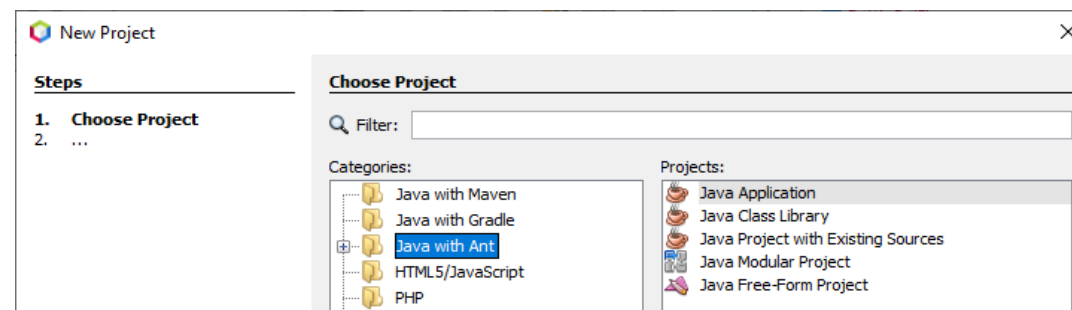
Introducción a los servicios Web en Java

Introducción

- Los *servicios Web* se implementan en aplicaciones Web



- Los *clientes* que consumen los *servicios Web* se pueden hacer desde cualquier tipo de proyecto:
 - Aplicaciones de consola
 - Aplicaciones Web



Introducción a los servicios Web en Java

Implementación cliente de servicios Web con JAX-WS: Ejemplo

Ejercicio: diseñar un cliente del servicio Web que se define en la página:

<http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso>

Que proporciona información de un país especificado por su código ISO

Introducción a los servicios Web en Java

Implementación cliente de servicios Web con JAX-RS: Ejemplo

Ejercicio: diseñar una aplicación web que tenga un enlace de acceso al *servicio Web* de *Google map* para presentar un mapa en una posición dada.

La *url* que necesitamos para acceder al servicio de mapas estáticos de Google es:

<https://maps.googleapis.com/maps/api/staticmap?parameters>

Algunos parámetros son: *center*, *zoom*, *size*, *format*, *markers*, *maptype*, *key* ...

<https://developers.google.com/maps/documentation/static-maps/intro>

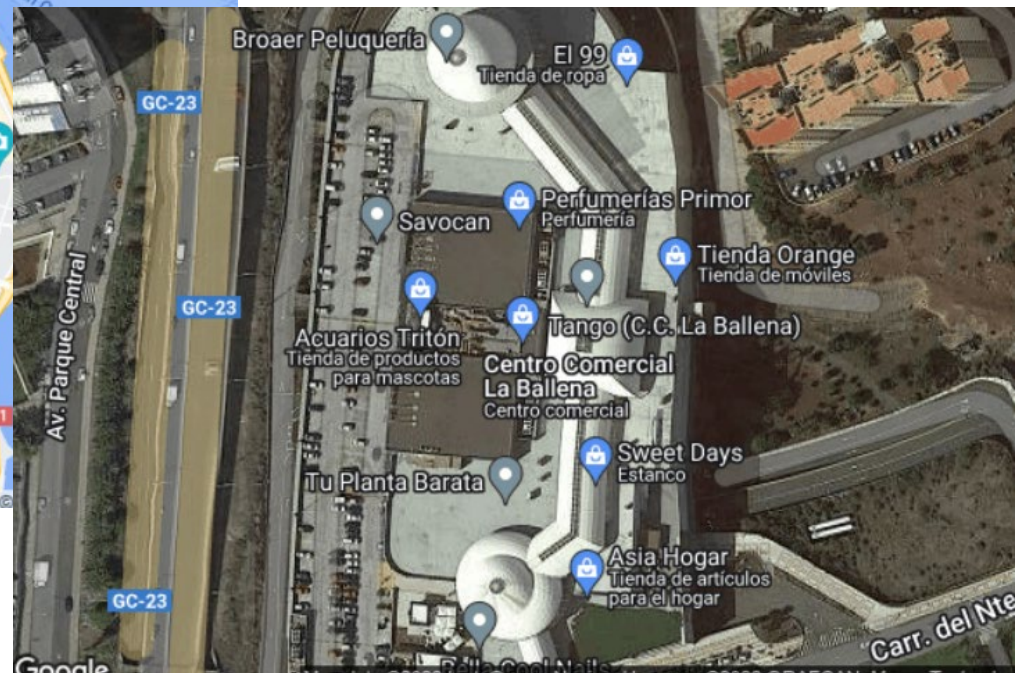
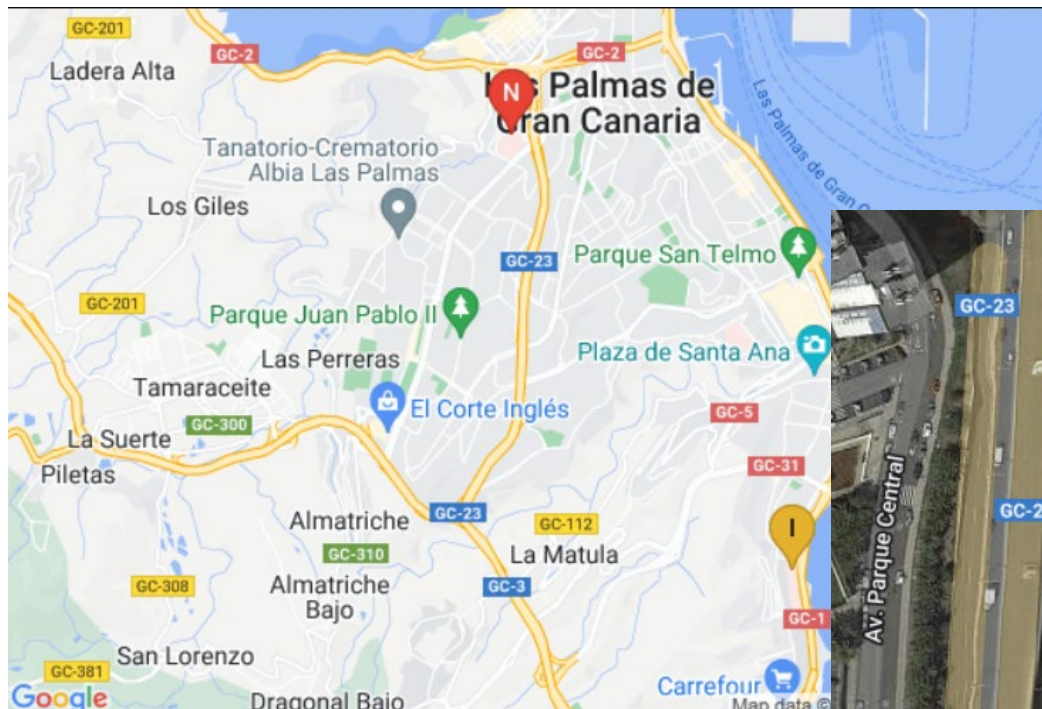


Se explica cómo obtener una clave

Introducción a los servicios Web en Java

Implementación cliente de servicios Web con JAX-RS: Ejemplo

https://maps.googleapis.com/maps/api/staticmap?center=Centro+Comercial+La+Ballena,+Las+Palmas&zoom=13&size=600x400&maptype=roadmap&markers=color:red|label:N|28.120833,-15.444722&markers=color:yellow|label:I|28.08379,-15.4175267&key=API_KEY



zoom=18 y maptype=hybrid

- Introducción a los servicios Web
 - Servicios web *SOAP*
 - Servicios web *RESTful*
- Introducción a los servicios Web en Java
 - JAX-WS y JAX-RS
 - Ejemplos de implementación de clientes de servicios Web
 - Ejercicios de implementación de servicios web

Introducción a los servicios Web en Java

JAX-WS

- **JAX-WS** (*Java API for XML Web Services*) permite desarrollar **servicios Web** basados en **SOAP**
- Utiliza **anotaciones** para el **manejo** de los **servicios Web**:
 - **@WebService** marca una **clase** que **implementa** un **servicio Web**. Debe tener un constructor sin parámetros
 - **@WebMethod** indica que un **método** es una **operación** del **servicio Web**. Estos métodos deben ser públicos
 - **@WebServiceRef** es usada para **definir** una **referencia** a un **servicio web** invocado por el cliente. Su atributo *wsdlLocation* indica la localización del documento WSDL
 - **@WebParam** se utiliza para **mapear** el **parámetro** **expuesto** en el **servicio web** con el utilizado por el método correspondiente

Introducción a los servicios Web en Java

Implementación de servicios Web con JAX-WS: Ejemplo

Ejercicio: diseñar un servicio Web SOAP con las siguientes operaciones:

- Calcular el ángulo en radianes de un ángulo en grados

$$angulo_radianes = \frac{angulo_grados \cdot \pi}{180}$$

- Indicar el tipo de ángulo (agudo, recto, obtuso)
- Calcular el ángulo suplementario de uno dado en grados

$$angulo_suplementario = 180 - angulo_grados$$

Diseñar un cliente del servicio Web anterior con dos páginas:

- **index.jsp:** página de inicio con un formulario donde lee el número, selecciona la operación deseada en una lista desplegable y lo envía a la página de salida mediante un botón de enviar
- **respuesta.java:** página de salida donde se presenta el resultado de la conversión

Introducción a los servicios Web en Java

Implementación de servicios Web con JAX-WS: Ejemplo



← → ↻ 🏠 ⓘ localhost:8080/CienteSV

Introduzca la info solicitada

Angulo en grados:

Convertir a radianes 

Enviar

1. Recuperar información de formulario en `index.jsp`
2. Llamar al servicio web
3. Generar página de salida



← → ↻ 🏠 ⓘ localhost:8080/CienteSWConversor/respues

56.0 grados se corresponde con 0.9773844 radianes

Diseñar un cliente del servicio Web anterior con dos páginas:

- ***index.jsp***: página de inicio con un formulario donde lee el número, selecciona la operación deseada en una lista desplegable y lo envía a la página de salida mediante un botón de enviar
- ***respuesta.java***: página de salida donde se presenta el resultado de la conversión

Introducción a los servicios Web en Java

JAX-RS

- *JAX-RS* (Java API for RESTful Web Services) permite desarrollar *servicios Web* basados en **REST**
- Utiliza anotaciones para el manejo de los *servicios Web*:
 - *@GET* indica que el método definido a continuación responderá a una petición **HTTP GET**. Existen también *@POST*, *@DELETE* y *@POST*
 - *@Produces(tipo)* indica el tipo de formato generado en las respuestas a una petición
 - *@Path(ruta)* indica la ruta de acceso correspondiente a una clase o método. Conformar la URI relativa a la URL del servidor
 - *@QueryParam* se utiliza para mapear el parámetro pasado por la URL con el utilizado por el método correspondiente

Introducción a los servicios Web en Java

Implementación de servicios Web con JAX-RS: Ejemplo

Ejercicio: diseñar un servicio Web RESTful que presente un mapa con la localización de una cafetería del campus universitario, a petición de un cliente.

El servicio Web debe tener un método llamado *getMapaCafeterias* que responde a peticiones GET y que produzca un resultado tipo *image/jpeg* (corresponde a una imagen *jpg*) en la que aparece el mapa con una marca situada sobre la cafetería. La URL que accede al servicio tiene el siguiente formato:

http://localhost:8080/ServicioMapaCampus/mapa/informacion/cafeterias?nombre=nombreCaf

