**Tema 2: REST – Servicios web en java**

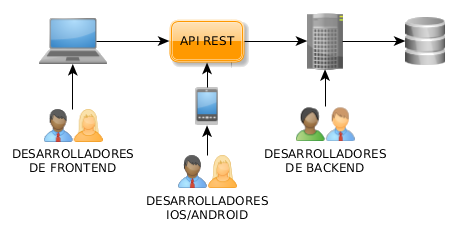
**¿Qué es API?**

las API son un conjunto de comandos, funciones y protocolos informáticos que permiten a los desarrolladores crear programas específicos para ciertos sistemas operativos. Las API simplifican en gran medida el trabajo de un creador de programas, ya que no tiene que escribir códigos desde cero. Estas permiten al informático usar funciones predefinidas para interactuar con el sistema operativo o con otro programa.

Para que sirven las API:

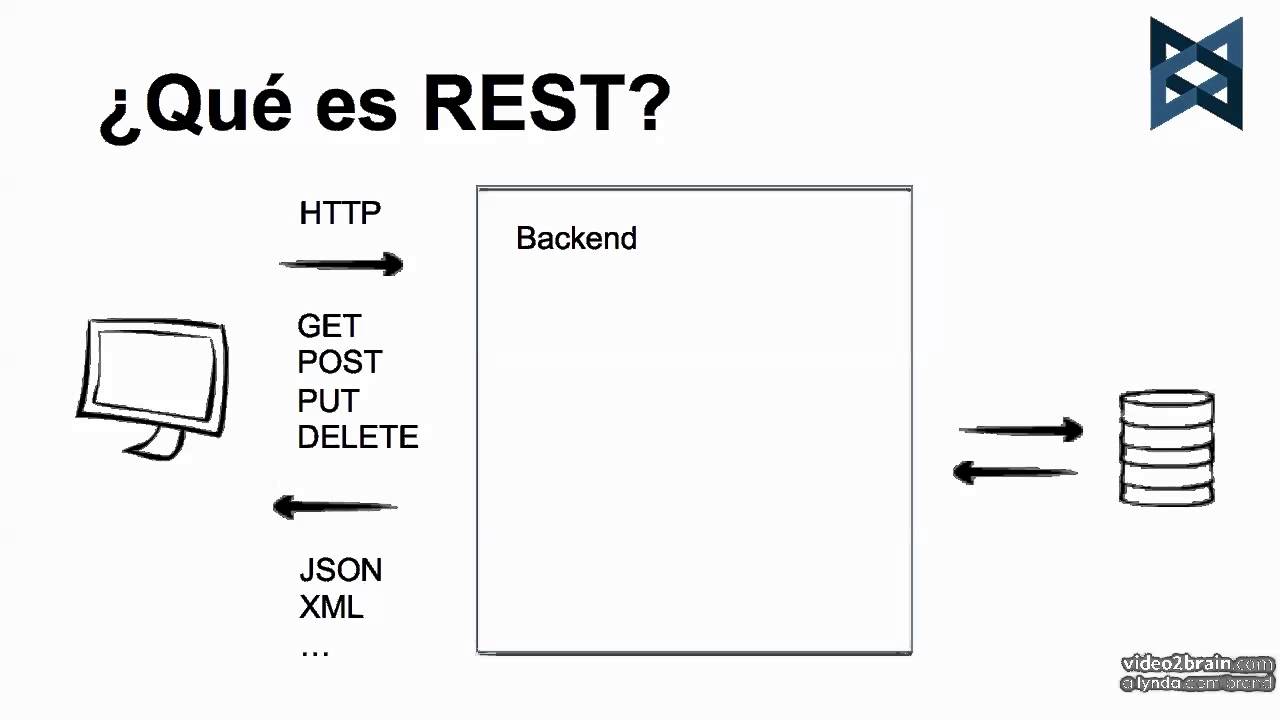
Las API pueden servir para comunicarse con el sistema operativo (WinAPI), con bases de datos (DBMS) o con protocolos de comunicaciones (Jabber/XMPP). En los últimos años, por supuesto, se han sumado múltiples redes sociales (Twitter, Facebook, YouTube, Flickr, LinkedIn, etc.) y otras plataformas online (Google Maps, WordPress…), lo que ha convertido el social media marketing es algo más sencillo, más rastreable y, por tanto, más rentable.

Ejemplo de cómo funcionan las API:



**¿Qué es REST?**

REST o también conocido como transferencia de estado representacional se refiere originalmente a un conjunto de principios de arquitectura, en la actualidad se usa en el sentido más amplio para describir cualquier interfaz entre sistemas que utilice directamente HTTP para obtener datos o indicar la ejecución de operaciones sobre los datos, en cualquier formato ([XML](https://es.wikipedia.org/wiki/XML), [JSON](https://es.wikipedia.org/wiki/JSON), etc.) sin las abstracciones adicionales de los protocolos basados en patrones de intercambio de mensajes, como por ejemplo [SOAP](https://es.wikipedia.org/wiki/SOAP). Es posible diseñar sistemas de servicios web de acuerdo con el estilo arquitectural REST de Fielding y también es posible diseñar interfaces [XMLHTTP](https://es.wikipedia.org/wiki/XMLHttpRequest) de acuerdo con el estilo de [llamada a procedimiento remoto](https://es.wikipedia.org/wiki/Remote_Procedure_Call) (RPC), pero sin usar SOAP. Estos dos usos diferentes del término *REST* causan cierta confusión en las discusiones técnicas, aunque [RPC](https://es.wikipedia.org/wiki/Remote_Procedure_Call) no es un ejemplo de REST.



**¿Qué es API REST?**

La api de rest, a la que también se conoce como API REST, es una interfaz de programación de aplicaciones que se apoya en la arquitectura REST para el desarrollo de aplicaciones en red. Aprovechando el lenguaje HTML, permite que cualquier empresa cree aplicaciones web sin problemas, aunque siempre en base a las restricciones que supone.

Las operaciones más importantes que nos permitirán manipular los recursos son cuatro: GET para consultar y leer, POST para crear, PUT para editar y DELETE para eliminar.

Una API REST define un conjunto de funciones que los desarrolladores pueden realizar solicitudes y recibir respuestas a través del protocolo HTTP, como GET y POST.

Debido a que la API REST usa HTTP, pueden ser utilizados por prácticamente cualquier lenguaje de programación y son fáciles de probar (es un requisito de una API REST que el cliente y el servidor sean independientes entre sí, lo que permite codificarlo en cualquier idioma y mejorar al soportar la longevidad y evolución)

Así funciona la API REST:



**Verbos en REST**

* **GET**

El método GET solicita una representación de un recurso específico. Las peticiones que usan el método GET sólo deben recuperar datos.

* **POST**

El método POST se utiliza para enviar una entidad a un recurso en específico, causando a menudo un cambio en el estado o efectos secundarios en el servidor

* **PUT**

El modo PUT reemplaza todas las representaciones actuales del recurso de destino con la carga útil de la petición.

* **DELETE**

El método DELETE borra un recurso en específico.

* **PATCH**

El método PATCH es utilizado para aplicar modificaciones parciales a un recurso.

**¿Qué es JSON?**

**JSON** es el acrónimo para JavaScript Object Notation, y aunque su nombre lo diga, no es necesariamente parte de JavaScript, de hecho es un estándar basado en texto plano para el intercambio de información, por lo que se usa en muchos sistemas que requieren mostrar o enviar información para ser interpretada por otros sistemas, la ventaja de JSON al ser un formato que es independiente de cualquier lenguaje de programación, es que los servicios que comparten información por éste método, no necesitan hablar el mismo idioma, es decir, el emisor puede ser Java y el receptor PHP, cada lenguaje tiene su propia librería para codificar y decodificar cadenas de JSON.

Ejemplo JSON para un objeto tipo persona:

{

"nombre":"Fulano Probencio",

"edad":27,

"nacionalidad":"chileno",

"altura":"172 cm",

"peso":75

}

**¿Qué es jersey en java?**

Jersey es un cliente Restful open-source, implementación de referencia de JAX-RS, el API de referencia para Web Service Restful. Este ultimo a su vez es un API de java que proporciona soporte a los web service creados bajo la arquitectura REST.

Ejemplo de jersey con rest en java:

**1-Introducir en el POM.xml las dependencias necesarias:**

<dependency>

<groupId>com.sun.jersey</groupId>

<artifactId>jersey-server</artifactId>

<version>1.19</version>

</dependency>

**2-En segundo lugar, crear la clase java que dará servicio REST en java, el soporte a los servicios de tipo REST:**

package prueba. proyecto;

import javax.ws.rs.GET;

import javax.ws.rs.Path;

import javax.ws.rs.PathParam;

import javax.ws.rs.core.Response;

@Path("/prueba") /\*URL a la que atenderá las peticiones\*/

public class RestService {

    @GET /\*Tipo de llamada\*/

    @Path("/{param}")

    public Response getMsg(@PathParam("param") String msg) {

        /\*Mostrará por pantalla el parámetro que le hemos pasado a la URL\*/

        String output = "My Jersey application says : " + msg;

        return Response.status(200).entity(output).build();

    }

}

**3-Configurar web.xml para indicar que cuando la URL cumpla determinado patrón, la petición será restringida a la clase que se acabó de crear:**

<?xml version="1.0" encoding="UTF-8"?>

<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app\_2\_5.xsd" version="2.5">

    <servlet>

        <servlet-name>Rest Servlet</servlet-name>

        <servlet-class>com.sun.jersey.spi.container.servlet.ServletContainer</servlet-class>

        <init-param>

            <param-name>com.sun.jersey.config.property.packages</param-name>

            <param-value>prueba.proyecto</param-value>

        </init-param>

        <load-on-startup>1</load-on-startup>

    </servlet>

    <servlet-mapping>

        <servlet-name>Rest Servlet</servlet-name>

        <url-pattern>/rest/\*</url-pattern>

    </servlet-mapping>

</web-app>

**Anotaciones jersey para el trabajo con REST**

Para construir un servicio REST utilizando los estándares de JAX-RS para ello lo primero que debemos conocer son las distintas anotaciones que el estándar define.

Las anotaciones básicas de JAX-RS son las siguientes:

@GET

@POST

@PUT

@DELETE

@Path

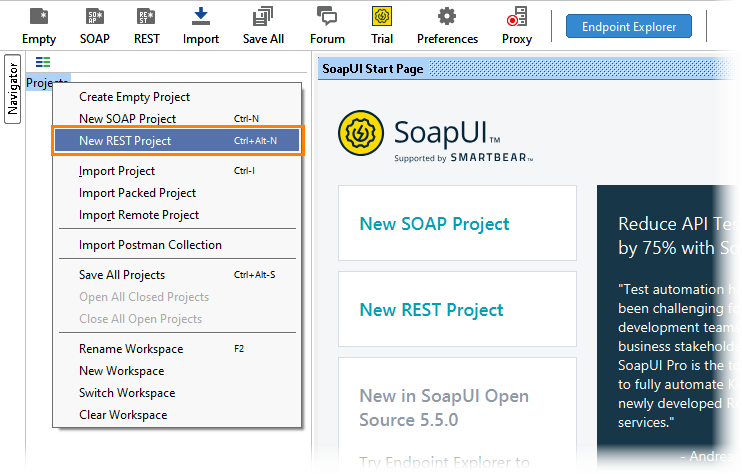
@Produces

**Uso de herramienta SoapUI para consumo de API rest**

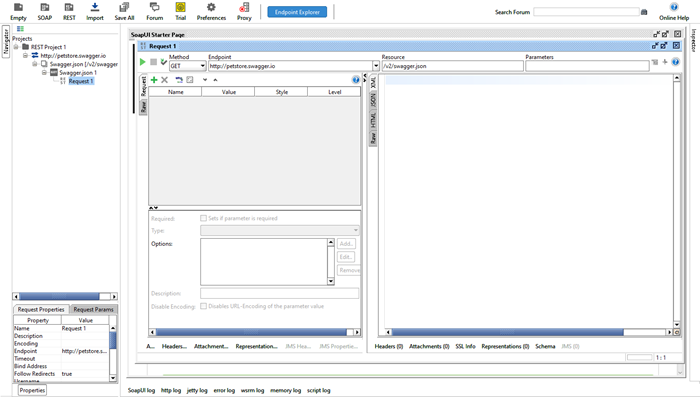
SoapUI es una herramienta de gran alcance diseñada para ayudar en la prueba y el desarrollo de aplicaciones. Permite efectuar el testeo de la web, con docenas de características, incluyendo una interfaz simple, fácil e intuitiva. Permite la utilización de métodos de captura y repetición, siendo una herramienta de gran ayuda en la realización de pruebas de carga de gran alcance, informes detallados, gráficos, etc...

Para trabajar SoapUI con API REST se deben realizar los siguientes pasos:

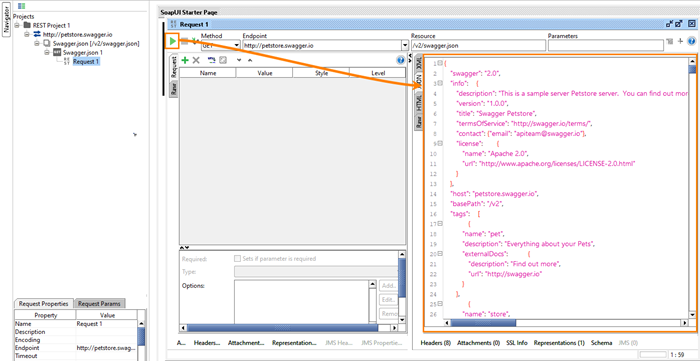
**Paso 1:** Después de abrir el programa SoapUI click derecho en New REST Project



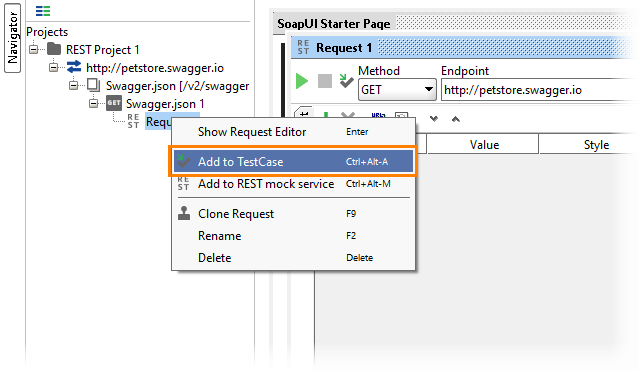
**Paso 2:** Aparecerá un cuadro de dialogo, luego en el cuadro de dialogo especifique la ruta URI de su API REST en el cuadro de edición URL.



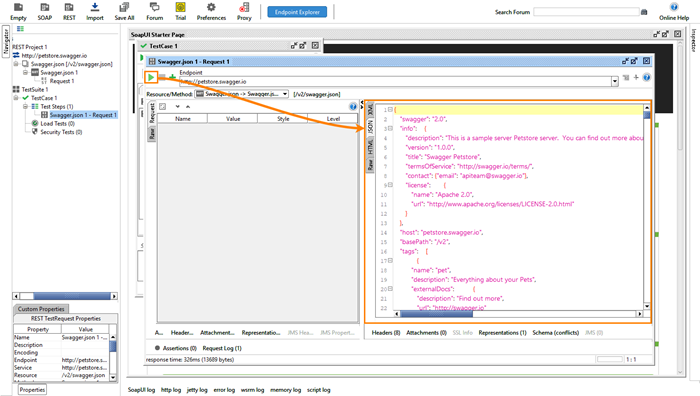
**Paso 3:** A continuación, se presentará la pantalla principal para proyectos REST. Haz click en el botón verde de reproducción en la esquina superior izquierda para ver la respuesta.



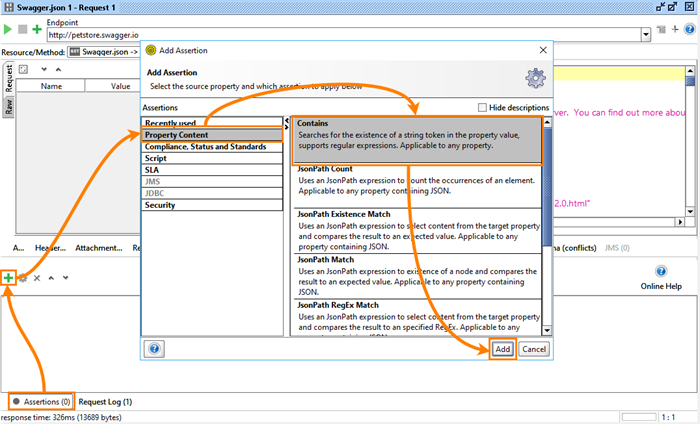
**Paso 4:** En la ventana del navegador izquierda haga click derecho en solicitud 1 y luego click en agregar caso de prueba.



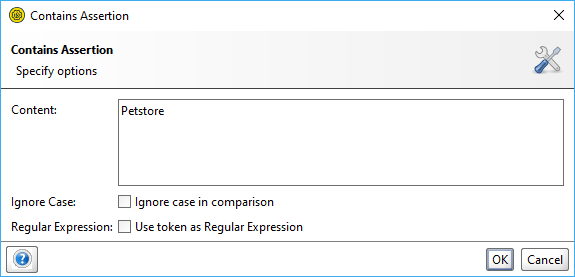
**Paso 5:** Hacer click nuevamente en el botón de reproducción para ver los resultados.



**Paso 6:** Para crear nuestra primera afirmación hacer click en la parte inferior izquierda y luego hacer click en signo más verde.



**Paso 7:** Seleccione contains assertions y verifique que petstore esté abierto.



**Creación cliente REST**

Para la creación de un cliente REST debemos de realizar los siguientes pasos:

**Paso 1:** Crear un proyecto Maven con varias dependencias



**Paso 2:** Crear la clase personas



**Paso 3:** Solicitar que nos devuelta la lista de personas es similar, pero hace falta apoyarse en genéricos.



**Paso 4:** El resultado nos mostrará una lista de elementos.

