

Patrón Estructural Composite



Universidad
del Cauca

Vigilada Mineducación

Laboratorio de Ingeniería de Software II

Practica de Laboratorio No. 8

Presentado por:

Jeferson Castaño Ossa

David Santiago Giron Muñoz

Profesor:

Ricardo Antonio Zambrano Segura

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Ingeniería de Sistemas

Popayán, Octubre de 2023

Arquitectura Cliente-Servidor

Capturas de pantalla de la ejecución de las aplicaciones:

1. EchoServerExample.

```
--- resources:3.3.1:resources (default-resources) @ strategyserver ---
skip non existing resourceDirectory C:\Users\IngSis\Documents\XD\client-server-evolution\strategyserver\src\main\resources

--- compiler:3.11.0:compile (default-compile) @ strategyserver ---
Nothing to compile - all classes are up to date

--- exec:3.1.0:exec (default-cli) @ strategyserver ---
nov 01, 2023 7:50:11 A.M. co.unicauca.strategyserver.infra.ServerSocketMultiThread openPort
INFO: Servidor iniciado, escuchando por el puerto 5.000
En servidor multihilo esta espera
```

2. AgencyTravelServer y AgencyTravelClient.

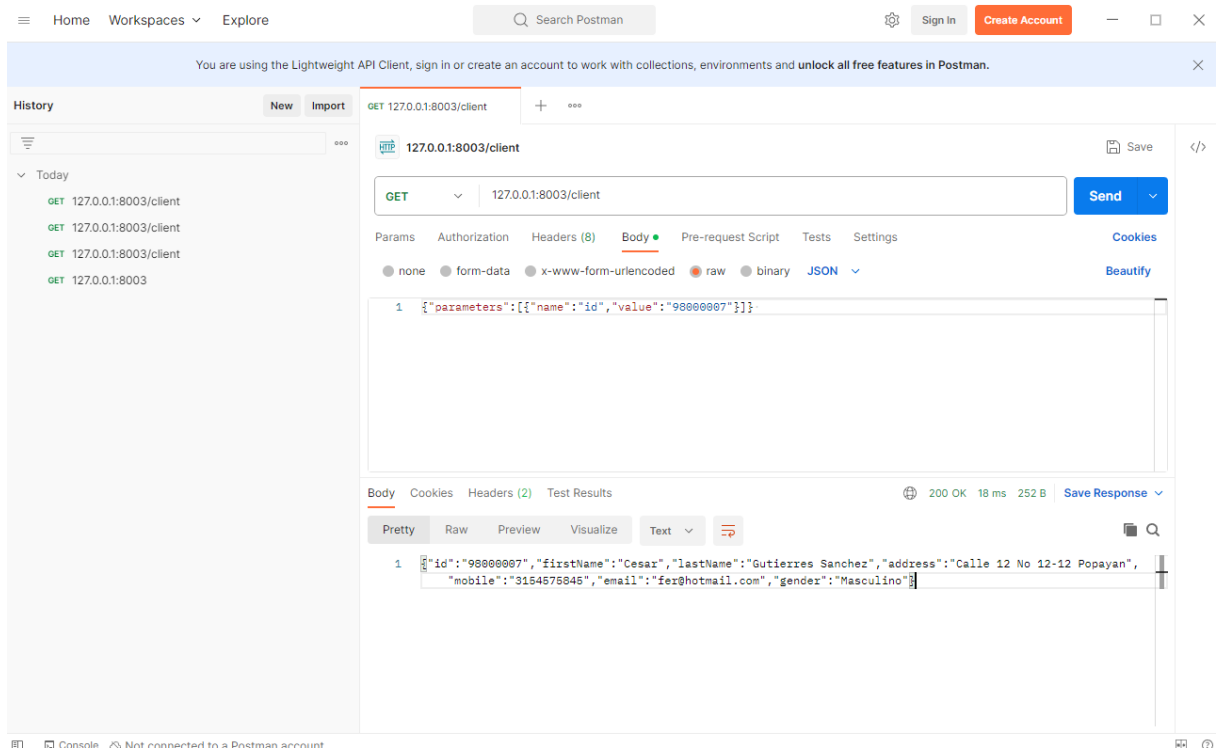
```
. skip non existing resourceDirectory C:\Users\IngSis\Documents\XD\client-server-evolution\TravelAgency-Server\src\main\resources

] --- compiler:3.11.0:compile (default-compile) @ TravelAgency-Server ---
. Nothing to compile - all classes are up to date

] --- exec:3.1.0:exec (default-cli) @ TravelAgency-Server ---
Ingrese el puerto de escucha
8003
nov 01, 2023 7:52:43 A.M. co.unicauca.strategyserver.infra.ServerSocketMultiThread openPort
INFO: Servidor iniciado, escuchando por el puerto 8.003
. En servidor multihilo esta espera
|
```

The screenshot shows a Java Swing window titled "Agencia de viajes". Inside, there's a menu bar with "Opciones", "Informes", "Configuracion", and "Ayuda (Invitado)". The main content area is titled "Consulta de Clientes" and contains the following text: "Este ejercicio aplica el patrón cliente/servidor. La aplicación cliente se conecta al servidor mediante Sockets. El servidor devuelve el objeto Cliente consultado en formato JSON. En el backend las cedulas desde 98000001 hasta 98000010." Below this text is a form with a label "Número de identificación:" followed by a text input field containing "98000001" and a "Buscar" button. Below the input field are several labels with corresponding text input fields: "*Nombres: Andrea", "*Apellidos: Sanchez", "Dirección: Calle 14 No 11-12 Popayan", "Celular: 3145878752", "Email: andrea@hotmail.com", and "*Sexo (M, F): Femenino". At the bottom of the form is a "Cerrar" button.

3. Consulta en el Web Server(infra.web) con postman.

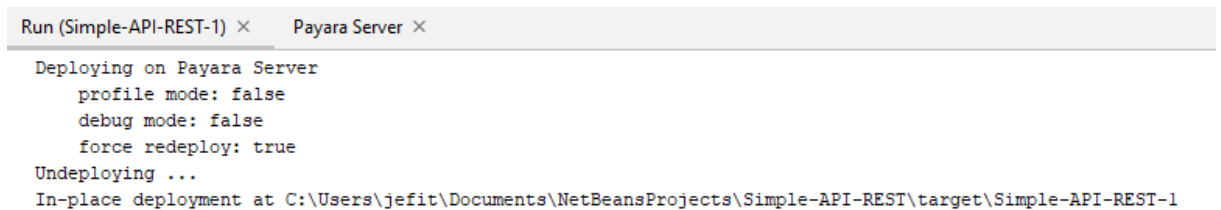


4. API - REST y cliente con Jersey Client API

Ejecución de Payara Server



Despliegue:





The screenshot shows a web browser window with the address bar displaying `localhost:8080/Simple-API-REST/product-service/products/`. The main content area shows a JSON array of three product objects. The first object has `id: 1`, `name: "Tv samsung"`, and `price: 200000`. The second object has `id: 2`, `name: "Tv lg"`, and `price: 300000`. The third object has `id: 1`, `name: "Tablet asus RESD-TD-34"`, and `price: 400000`. The JSON is formatted with syntax highlighting.

```
1 [
2   {
3     "id": 1,
4     "name": "Tv samsung",
5     "price": 200000
6   },
7   {
8     "id": 2,
9     "name": "Tv lg",
10    "price": 300000
11  },
12  {
13    "id": 1,
14    "name": "Tablet asus RESD-TD-34",
15    "price": 400000
16  }
17 ]
```

Ejecución del cliente Jersey:

```
public static void main(String[] args) {
    // CREANDO UN PRODUCTO
    NewJerseyClient jersey = new NewJerseyClient();
    //POST
    String rta = jersey.create_JSON(new Product(id: 5, name:"Nevera Lg", price:4000000d));
    System.out.println("Rta " + rta);
    // BUSCANDO UN PRODUCTO
    //GET
    Product product = jersey.findById(responseType: Product.class, id: "1");
    System.out.println(x: product.getId());
    System.out.println(x: product.getName());
    System.out.println(x: product.getPrice());
    // PROBAR LOS DEMAS SERVICIOS
    //REMOVE
    rta = jersey.remove(id: "1");
    System.out.println("Rta " + rta);
    //PUT
    rta = jersey.update_JSON(new Product(id: 5, name:"Patineta", price:40500d));
    System.out.println("Rta " + rta);

    //GET ALL

    Product[] products = jersey.findAll(responseType: Product[].class);
    for(int i = 0; i < products.length; i++){
        System.out.println(products[i].getId() + " " + products[i].getName() + " " + products[i].getPrice());
    }
}
```

```
-----< co.unicauca.products:Cliente-Rest >-----
Building Cliente-Rest 1
-----[ jar ]-----

--- exec-maven-plugin:3.1.0:exec (default-cli) @ Cliente-Rest ---
Rta {"ok":"true", "mensaje":"Producto creado","errores":""}
1
Tv samsung
200000.0
Rta {"ok":"true", "mensaje":"Producto borrado","errores":""}
Rta {"ok":"true", "mensaje":"Producto modificado","errores":""}
2 Tv lg 300000.0
1 Tablet asus RESD-TD-34 400000.0
5 Patineta 40500.0
-----
BUILD SUCCESS
-----
Total time:  1.972 s
Finished at: 2023-11-03T20:39:50-05:00
-----
```

Proyecto del servidor y cliente se encuentran en el repositorio.