

Microservicios – Arquitectura y Desarrollo

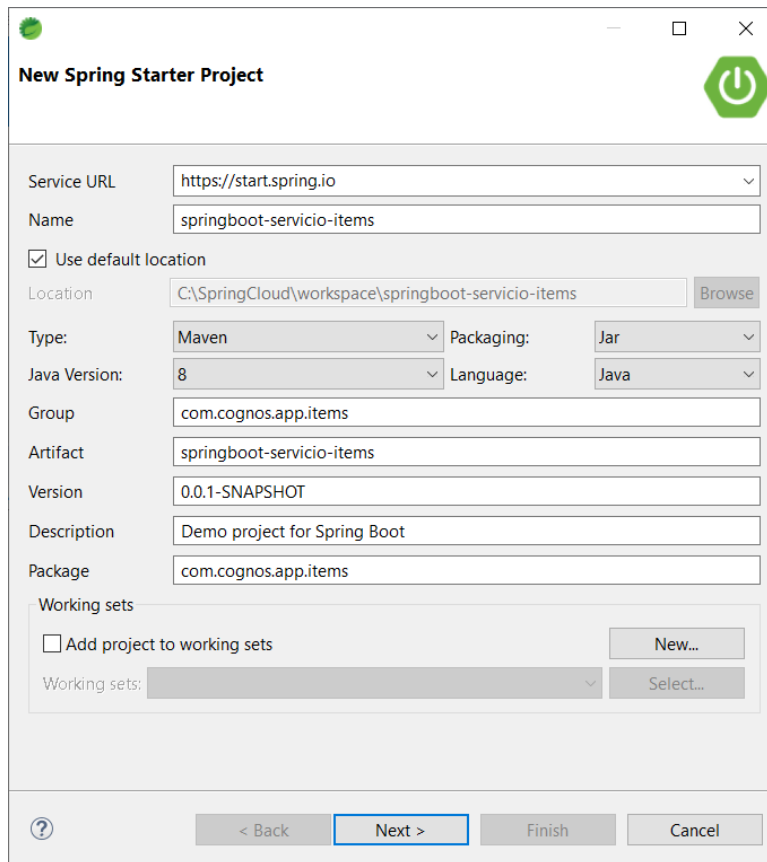
LAB Interacción entre Microservicios

Objetivos

- Mostrar al participante el procedimiento para el balanceo de carga de Microservicios

Procedimiento

1. Crea el proyecto **springboot-servicio-item** con las siguientes dependencias.



The screenshot shows the 'New Spring Starter Project' dialog box in the Eclipse IDE. The dialog is titled 'New Spring Starter Project' and features a green power button icon in the top right corner. The fields are filled with the following information:

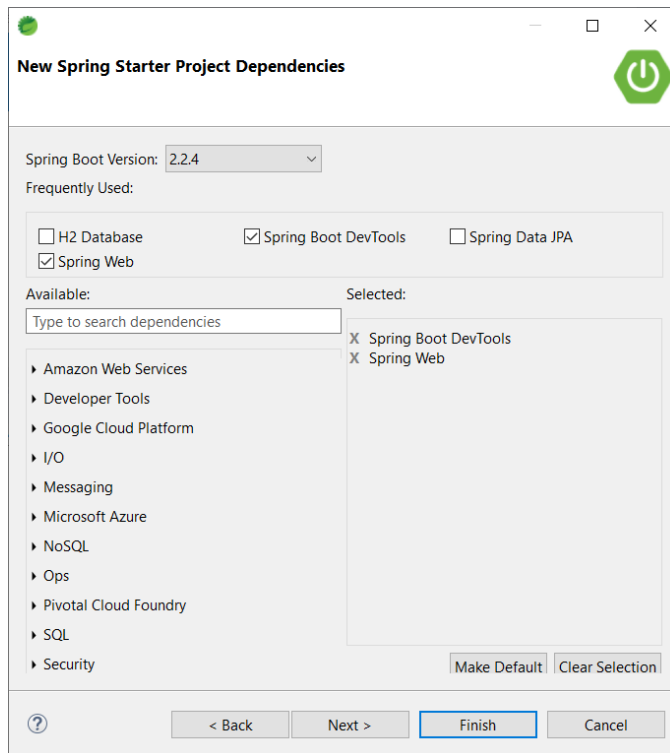
- Service URL: `https://start.spring.io`
- Name: `springboot-servicio-items`
- ☒ Use default location
- Location: `C:\SpringCloud\workspace\springboot-servicio-items` (with a 'Browse' button)
- Type: `Maven` (dropdown)
- Packaging: `Jar` (dropdown)
- Java Version: `8` (dropdown)
- Language: `Java` (dropdown)
- Group: `com.cognos.app.items`
- Artifact: `springboot-servicio-items`
- Version: `0.0.1-SNAPSHOT`
- Description: `Demo project for Spring Boot`
- Package: `com.cognos.app.items`

Below the main fields is a 'Working sets' section with the following options:

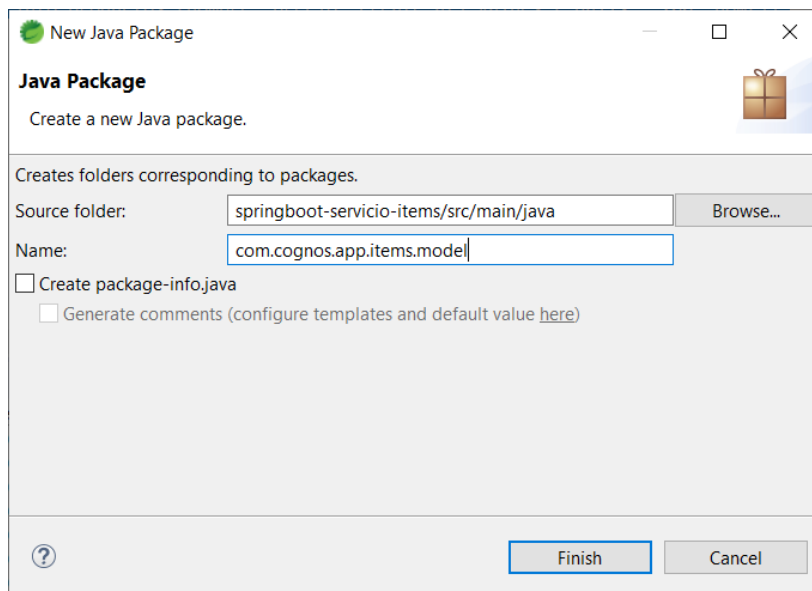
- ☐ Add project to working sets (with a 'New...' button)
- Working sets: (dropdown menu) (with a 'Select...' button)

At the bottom of the dialog, there are navigation buttons: a help icon (?), '< Back', 'Next >' (highlighted with a blue border), 'Finish', and 'Cancel'.

2. Selecciona las dependencias **“Spring Web”** y **“Spring Boot DevTools”**



3. Crea el paquete `com.cognos.app.items.model`



4. En el paquete `com.cognos.app.items.model` crea las clases `Producto` e `Item`

```
package com.cognos.app.items.model;  
  
import java.util.Date;
```

```

public class Producto {
    private Long id;
    private String nombre;
    private Double precio;
    private Date createdAt;
    public Long getId() {
        return id;
    }
    public void setId(Long id) {
        this.id = id;
    }
    public String getNombre() {
        return nombre;
    }
    public void setNombre(String nombre) {
        this.nombre = nombre;
    }
    public Double getPrecio() {
        return precio;
    }
    public void setPrecio(Double precio) {
        this.precio = precio;
    }
    public Date getCreatedAt() {
        return createdAt;
    }
    public void setCreatedAt(Date createdAt) {
        this.createdAt = createdAt;
    }
}

```

```

package com.cognos.app.items.model;

public class Item {
    private Producto producto;
    private Integer cantidad;

    public Item(Producto producto, Integer cantidad) {
        super();
        this.producto = producto;
        this.cantidad = cantidad;
    }
    public Item() {
    }
    public Producto getProducto() {
        return producto;
    }
    public void setProducto(Producto producto) {
        this.producto = producto;
    }
}

```

```

    public Integer getCantidad() {
        return cantidad;
    }
    public void setCantidad(Integer cantidad) {
        this.cantidad = cantidad;
    }
    public Double getTotal() {
        return producto.getPrecio()*cantidad.doubleValue();
    }
}

```

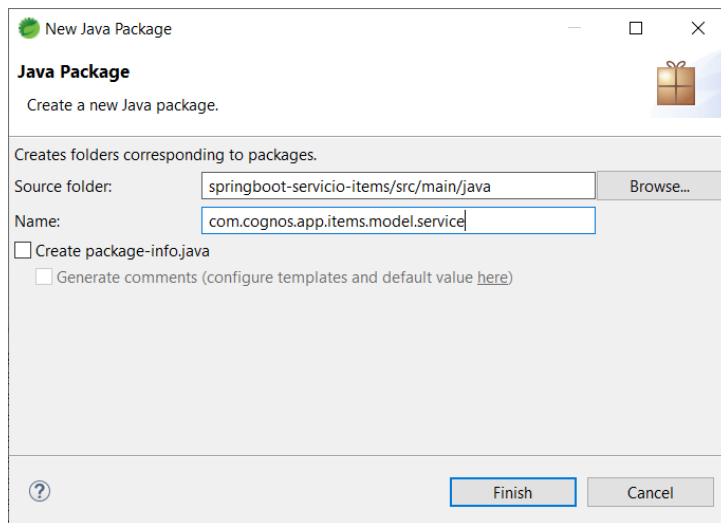
5. Configura el nombre y puerto del servicio springboot-servicio-items en application.properties

```

spring.application.name=servicio-items
server.port=8082

```

6. Crea el Paquete com.cognos.app.items.model.service



7. Crea la interface ItemService en el Paquete com.cognos.app.items.model.service

```

package com.cognos.app.items.model.service;

```

```

import java.util.List;

```

```

import com.cognos.app.items.model.Item;

```

```

public interface ItemService {

    public List<Item> findAll();

    public Item findById(Long id, Integer cantidad);

}

```

8. En el paquete com.cognos.app.items crea la clase AppConfig para registrar un cliente Rest

```

package com.cognos.app.items;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.client.RestTemplate;

@Configuration
public class AppConfig {
    @Bean("clienteRest")
    public RestTemplate registrarRestTemplate() {
        return new RestTemplate();
    }
}

```

9. Crea la clase ItemServiceImpl en el paquete com.cognos.app.items.model.service

```

package com.cognos.app.items.model.service;

import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.stream.Collectors;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import org.springframework.web.client.RestTemplate;

import com.cognos.app.items.model.Item;
import com.cognos.app.items.model.Producto;

@Service
public class ItemServiceImpl implements ItemService {

    @Autowired
    private RestTemplate clienteRest;

    @Override
    public List<Item> findAll() {
        String url="http://localhost:8081/listar";
    }
}

```

```

        List<Producto> productos =
Arrays.asList(clienteRest.getForObject(url, Producto[].class));
        List<Item> items = productos.stream().map(p -> new
Item(p,1)).collect(Collectors.toList());
        return items;
    }

    @Override
    public Item findById(Long id, Integer cantidad) {
        Map<String, String> pathVariables = new HashMap<String,String>();
        pathVariables.put("id",id.toString());
        String url = "http://localhost:8081/mostrar/{id}";
        Producto producto = clienteRest.getForObject(url,
Producto.class,pathVariables);
        return new Item(producto, cantidad);
    }
}

```

10. Crea el paquete com.cognos.app.items.controllers y la clase ItemController

```

package com.cognos.app.items.controllers;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RestController;

import com.cognos.app.items.model.Item;
import com.cognos.app.items.model.service.ItemService;

@RestController
public class ItemController {
    @Autowired
    private ItemService itemService;

    @GetMapping("/listar")
    public List<Item> listar(){
        return itemService.findAll();
    }

    @GetMapping("/ver/{id}/cantidad/{cantidad}")
    public Item detalle(@PathVariable Long id,@PathVariable Integer cantidad) {
        return itemService.findById(id, cantidad);
    }
}

```

11. Inicia los dos microservicios y prueba los servicios con postman

Postman

File Edit View Help

New Import Runner My Workspace Invite

Launchpad GET http://localhost:8082/mostrar/2/cantidad/7 Bootcamp No Environment

History Collections APIs BETA

Save Responses Clear all

Today

- GET http://localhost:8082/mostrar/2/cantidad/7
- GET http://localhost:8082/listar
- GET http://localhost:8081/listar
- GET http://localhost:8081/listar

Untitled Request

GET http://localhost:8082/mostrar/2/cantidad/7 Send Save

Params Authorization Headers (7) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (5) Test Results Status: 200 OK Time: 87ms Size: 289 B Save Response

Pretty Raw Preview Visualize BETA JSON

```
1 {
2   "producto": {
3     "id": 2,
4     "nombre": "T-Shirt",
5     "precio": 500.0,
6     "createAt": "2020-01-30T00:00:00.000+0000"
7   },
8   "cantidad": 7,
9   "total": 3500.0
10 }
```

Bootcamp Build Browse