**Spring Cloud Netflix Ribbon** 🧭  
It used to be a **client-side load balancer** widely used in microservices (especially with Eureka + Feign), but note:

⚠️ **Ribbon is now deprecated** in Spring Cloud and replaced by **Spring Cloud LoadBalancer**.  
Still, let’s cover Ribbon completely since many existing projects use it.

**🧠 1️⃣ What is Ribbon?**

Ribbon is a **client-side load balancer**:

* It runs inside your service.
* When your service calls another service (e.g., Currency Conversion → Currency Exchange), Ribbon picks a server instance from the list and forwards the request.
* Works great with **Eureka** — Ribbon auto-fetches instances registered with Eureka.

**🏗 2️⃣ Ribbon Architecture**

Currency-Conversion-Service

|

| Ribbon + RestTemplate / Feign

v

Eureka (Service Registry) ----> Multiple instances of Currency-Exchange-Service

* Ribbon decides which instance to call, using a **load-balancing algorithm** (e.g., round robin by default).

**⚙️ 3️⃣ Add Ribbon Dependency**

If you are not using Feign (or old Spring Cloud starter already includes it), add:

<dependency>

<groupId>org.springframework.cloud</groupId>

<artifactId>spring-cloud-starter-netflix-ribbon</artifactId>

</dependency>

But if you use **spring-cloud-starter-openfeign**, Ribbon is included automatically in older Spring Cloud versions.

**📝 4️⃣ Ribbon with Eureka**

You don’t need to configure anything extra. Just register both services in Eureka:

**✅ Currency Exchange Service**

spring.application.name=currency-exchange-service

server.port=8000

eureka.client.service-url.default-zone=http://localhost:8761/eureka

**✅ Currency Conversion Service**

spring.application.name=currency-conversion-service

server.port=8100

eureka.client.service-url.default-zone=http://localhost:8761/eureka

Then, in Conversion service:

@LoadBalanced // Ribbon-enabled RestTemplate

@Bean

public RestTemplate restTemplate() {

return new RestTemplate();

}

@Autowired

private RestTemplate restTemplate;

public String callExchangeService() {

// Notice: Using service name instead of hostname

String url = "http://currency-exchange-service/exchange/from/USD/to/INR";

return restTemplate.getForObject(url, String.class);

}

✅ Ribbon + Eureka automatically load balances across all running instances of currency-exchange-service.

**🧠 5️⃣ Ribbon Without Eureka**

You can also define static servers manually:

spring:

cloud:

loadbalancer:

ribbon:

enabled: true # optional in older versions

currency-exchange-service:

ribbon:

listOfServers: localhost:8000,localhost:8001

or in .properties:

currency-exchange-service.ribbon.listOfServers=localhost:8000,localhost:8001

Then you can still use:

restTemplate.getForObject("http://currency-exchange-service/exchange", String.class);

Ribbon will round-robin between 8000 and 8001.

**🧪 6️⃣ Ribbon Load Balancing Rules**

Ribbon uses IRule to decide which server to call.

Some built-in rules:

| **Rule** | **Behavior** |
| --- | --- |
| RoundRobinRule | Default — cycles through servers |
| RandomRule | Picks a server randomly |
| WeightedResponseTimeRule | Prefers faster servers |
| BestAvailableRule | Picks server with fewest concurrent requests |

Example to change rule:

@Configuration

@RibbonClient(name = "currency-exchange-service", configuration = RibbonConfig.class)

public class CurrencyExchangeRibbonConfig {

}

@Configuration

public class RibbonConfig {

@Bean

public IRule ribbonRule() {

return new RandomRule(); // or RoundRobinRule()

}

}

**🧠 7️⃣ Ribbon + Feign**

If you use **Feign**, Ribbon works automatically underneath:

@FeignClient(name = "currency-exchange-service")

public interface CurrencyExchangeProxy {

@GetMapping("/exchange/from/{from}/to/{to}")

ExchangeValue getExchangeValue(@PathVariable String from, @PathVariable String to);

}

No URL needed — Ribbon + Eureka resolve it.

**⚠️ 8️⃣ Ribbon Deprecation**

* Ribbon has been **deprecated** since **Spring Cloud Hoxton**.
* Replacement: **Spring Cloud LoadBalancer**.
  + Works similarly but is independent of Netflix OSS.
  + Automatically works with Feign and RestTemplate.

New way:

@LoadBalanced

@Bean

public RestTemplate restTemplate() {

return new RestTemplate();

}

And no extra Ribbon dependency needed.

**✅ Summary**

| **Feature** | **Ribbon 🧭** |
| --- | --- |
| Purpose | Client-side load balancing |
| Works with | Eureka, Feign, RestTemplate |
| Configuration | \*.ribbon.listOfServers or Eureka |
| Load balancing | Round robin by default |
| Status | ❌ Deprecated → Use LoadBalancer |
| Replacement | Spring Cloud LoadBalancer |

**🚀 Example:**

@LoadBalanced

@Bean

public RestTemplate restTemplate() {

return new RestTemplate();

}

public String getExchangeValue() {

return restTemplate.getForObject(

"http://currency-exchange-service/exchange/from/USD/to/INR",

String.class

);

}

✅ Ribbon will pick one instance automatically.