DGTIC UNAM
DIRECCIÓN GENERAL DE CÓMPUTO Y DE TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN

REDEC UNAM

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
DIRECCIÓN GENERAL DE CÓMPUTO Y DE TECNOLOGÍAS
DE INFORMACIÓN Y COMUNICACIÓN

DIPLOMADO

Desarrollo de sistemas con tecnología Java

Módulo 10

API RESTful con Spring Boot

M. en C. Jesús Hernández Cabrera

Consumiendo una API REST con Spring Boot MVC (Web)

- **Spring Boot** ofrece dos enfoques principales para consumir APIs REST:
  - **RestTemplate**: Una clase sincronizada y fácil de usar para peticiones HTTP.
  - **WebClient**: Un cliente reactivo, recomendado para aplicaciones no bloqueantes y asíncronas.

RestTemplate

- RestTemplate es una clase de Spring que facilita la comunicación con servicios REST.
- **Sencillo de usar**
  - Soporta métodos HTTP (GET, POST, PUT, DELETE).
- **Pros**
  - Ideal para llamadas síncronas y más fácil de implementar en aplicaciones simples.
- **Contras**
  - Obsoleto para aplicaciones no bloqueantes; WebClient es el sucesor.

Dependencia

```
  <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
  </dependency>

<dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-thymeleaf</artifactId>
</dependency>
```

Ejemplo

```
RestTemplate restTemplate = new RestTemplate();
String url = "http://mi.sitio.com/data";
ResponseEntity<String> response =
restTemplate.getForEntity(url, String.class);
```

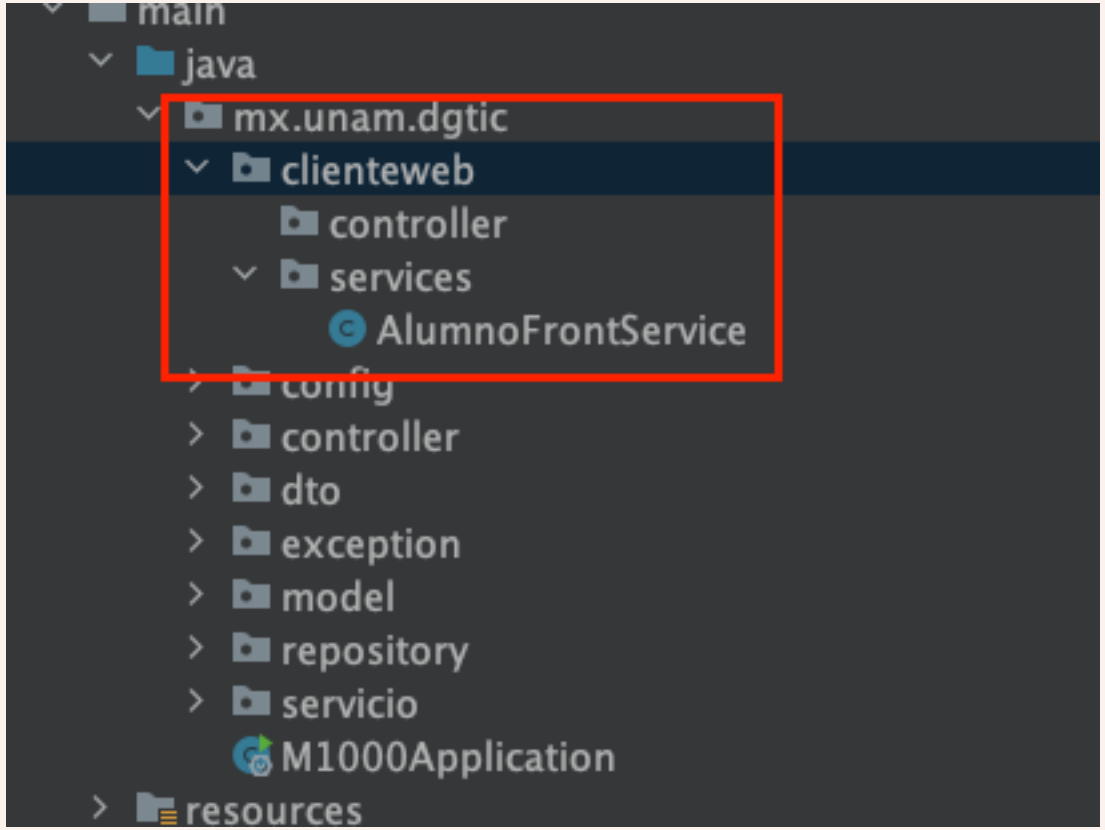
Métodos de RestTemplate

- **GET**
  - getForObject() o getForEntity() - Obtiene datos del servidor.
- **POST**
  - postForObject() o postForEntity() - Envía datos al servidor.
- **PUT**
  - put() - Actualiza un recurso existente.
- **DELETE**
  - delete() - Elimina un recurso del servidor.

Configurar RestTemplate con Spring Boot

```
@Configuration public class AppConfig {
        @Bean public RestTemplate restTemplate() {
                return new RestTemplate();
        }
}
```

Demo en la misma Aplicación



Ejemplo GET RestTemplate

```
RestTemplate restTemplate = new RestTemplate();
Alumno alumno =
restTemplate.getForObject("http://localhost:8080/api/v2/alumnos/{id}",
Alumno.class, id);
```

Consumir

```java
@Controller
public class AlumnoFrontController {

    @Autowired
    private AlumnoFrontService alumnoFrontService;

    @GetMapping("/front/alumnos/{matricula}")
    public String getAlumno(@PathVariable String matricula, Model model) {
        AlumnoDto alumno = alumnoFrontService.getAlumnoByMatricula(matricula);
        model.addAttribute("alumno", alumno);
        return "alumnodetalle";
    }

}
```

DGTIC UNAM
DIRECCIÓN GENERAL DE CÓMPUTO Y DE TECNOLOGÍAS DE INFORMACIÓN Y COMUNICACIÓN

REDEC UNAM

UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO
DIRECCIÓN GENERAL DE CÓMPUTO Y DE TECNOLOGÍAS
DE INFORMACIÓN Y COMUNICACIÓN

DIPLOMADO

Desarrollo de sistemas con tecnología Java

WebClient

WebClient

- WebClient es el cliente HTTP moderno en Spring WebFlux.
- **Asíncrono y Reactivo**
  - Diseñado para operaciones no bloqueantes.
- **Recomendado** para aplicaciones con un alto rendimiento y concurrencia.

```
WebClient webClient = WebClient.create("http://api.example.com");
Mono<String> response = webClient.get()
                                 .uri("/data")
                                 .retrieve()
                                 .bodyToMono(String.class);
```

Métodos de WebClient

- **GET**: ***.get()***
  - Para obtener recursos.
- **POST**: ***.post()***
  - Para enviar datos.
- **PUT**: ***.put()***
  - Para actualizar datos.
- **DELETE**: ***.delete()***
  - Para eliminar recursos.

Dependencia

```
<dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-webflux</artifactId>
</dependency>
```

Ejemplo GET WebClient

```
WebClient webClient = WebClient.create("http://api.example.com");
Mono<Alumno> alumnoMono = webClient.get()
                                .uri("/alumnos/{id}", id)
                                .retrieve()
                                .bodyToMono(Alumno.class);
```

Configuración

```
@Configuration
public class WebClientConfig {
    @Bean
    public WebClient webClient() {
        return WebClient.builder()
                        .baseUrl("http://api.example.com")
                        .build();
    }
}
```

Característica	RestTemplate	WebClient
Bloqueante	Sí	No
Soporte Reactivo	No	Sí
Uso recomendado	Apps Síncronas	Apps Asíncronas
Futuro de Spring	Dejará de usarse	Recomendado

Contacto

M. En C. Jesús Hernández Cabrera

Profesor de carrera

jesushc@unam.mx

Redes sociales:

www.linkedin.com/in/hcjesus