



universidad
de león



Escuela de Ingenierías
Industrial, Informática y
Aeroespacial

GRADO EN INGENIERÍA INFORMÁTICA

Trabajo de Fin de Grado

TÍTULO DEL TRABAJO

TITLE OF THE WORK

Autor: Samuel Castrillo Domínguez
Tutor: Eva María Cuervo Fernández

Junio, 2023

UNIVERSIDAD DE LEÓN
Escuela de Ingenierías Industrial,
Informática y
Aeroespacial

GRADO EN INGENIERÍA
INFORMÁTICA
Trabajo de Fin de Grado

ALUMNO: Samuel Castrillo Domínguez

TUTOR: Eva María Cuervo Fernández

TÍTULO: Título del trabajo

TITLE: Title of the work

CONVOCATORIA: Junio, 2023

RESUMEN:

El resumen reflejará las ideas principales de cada una de las partes del trabajo, pudiendo incluir un avance de los resultados obtenidos. Constará de un único párrafo y se recomienda una longitud no superior a 300 palabras. En cualquier caso, no deberá superar una página de longitud.

ABSTRACT:

Abstract will reflect the main ideas of each part of the work, including an advance of the results obtained. It will consist of a single paragraph and it is recommended a length not superior to 300 words. In any case, it should not exceed a page of length.

Palabras clave: Lorem, ipsum, dolor, sit, amet.

Firma del alumno:

VºBº Tutor/es:

Índice de contenidos

Índice de figuras	II
Índice de cuadros y tablas	III
Índice de bloques de código	IV
1. Introducción	1
2. Contenido	2
2.1. Patrones de diseño	2
2.1.1. Builder	2
2.1.2. Singleton	3
2.1.3. Strategy	4
Bibliografía	7

Índice de figuras

Índice de cuadros y tablas

2.1. Relación entre método HTTP y ruta.	3
---	---

Índice de bloques de código

- 2.1. Rutas utilizadas para simplificar el proceso de autorización de los usuarios. 4

1. Introducción

2. Contenido

2.1. PATRONES DE DISEÑO

Un patrón de diseño es una solución que se puede aplicar a diferentes contextos y que se puede reutilizar en diferentes proyectos.

Hay varias categorías de patrones de diseño, cada una con una finalidad diferente:

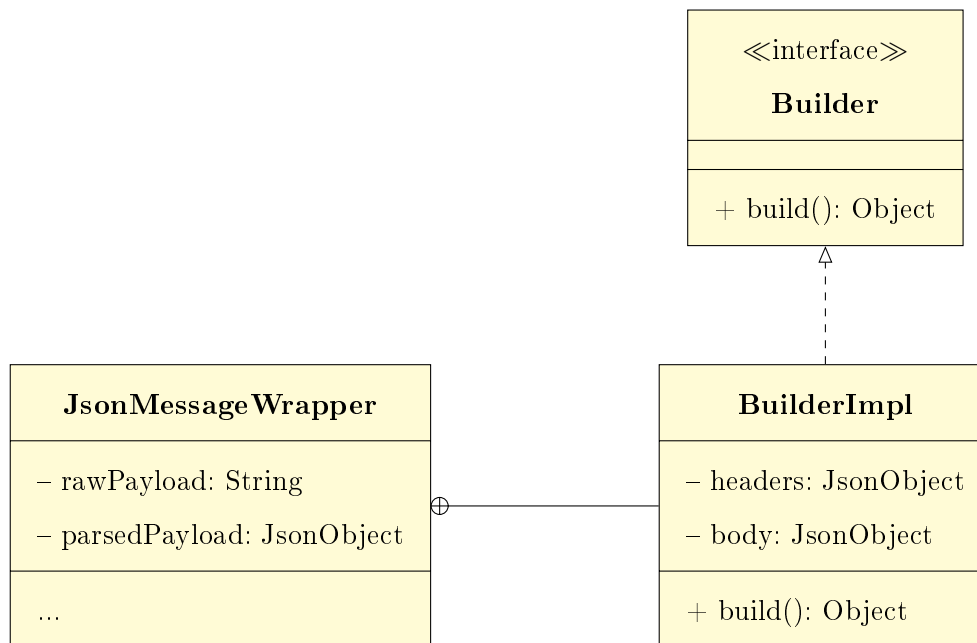
- **Patrones de creación:** se utilizan para crear objetos.
- **Patrones estructurales:** se utilizan para estructurar objetos.
- **Patrones de comportamiento:** se utilizan para definir la interacción entre objetos.

En este proyecto se han utilizado varios patrones de diseño, para permitir una mejor escalabilidad, mantenibilidad y reutilización del código. A continuación se detalla cierta información de los patrones utilizados:

- Definición / categoría.
- Explicación de cómo se ha implementado en el proyecto.
- Diagrama UML.
- Justificación de su uso en la aplicación.

2.1.1. BUILDER

Es un patrón de **creación** que permite instanciar objetos complejos de una forma sencilla. Se ha creado una interfaz **Builder** genérica para su reutilización en caso de ser necesaria. Esta interfaz define el método `build()` que devolverá la instancia de un objeto con los datos establecidos previamente.



En este proyecto se ha usado para simplificar la creación de objetos de tipo `JsonMessageWrapper`, por el momento. Esta clase es la encargada de encapsular los mensajes que se envían a través de la red en formato JSON.

2.1.2. SINGLETON

Este patrón se utiliza para garantizar que una clase tenga una única instancia y proporciona un punto de acceso global a ella [1]. En el contexto de esta aplicación, se utiliza en ciertas clases de utilidad y en las clases que asocian rutas a un método HTTP (por ejemplo `/login` con el método POST). Estas últimas son clases internas de `Routes.java` y los nombres dependen del método HTTP que se debe utilizar para realizar una petición a una ruta específica.

Cuadro 2.1: Relación entre método HTTP y ruta.

Método HTTP	Clase de la ruta
GET	<code>GetRoute</code>
POST	<code>PostRoute</code>
PUT	<code>PutRoute</code>
DELETE	<code>DeleteRoute</code>

Se han utilizado diferentes formas de acceso a la instancia de las clases. En el caso de las clases que asocian rutas a métodos HTTP, el modificador de acceso a la instancia es público. En otros casos, se provee un método estático para obtener la instancia de la clase, como se muestra en el siguiente ejemplo:

```
public class Routes {
    private Routes() {}
    ...
    public static class GetRoute {
        public static final GetRoute INSTANCE = new GetRoute();

        private GetRoute() {}

        public static final String USERS_URL = V_1 + "/users";

        /* Array containing the routes allowed by the low tier user */
        public String[] lowTierRoutes() {...}

        /* Array containing the routes allowed by the medium tier user */
        public String[] mediumTierRoutes() {...}

        /* Array containing the routes allowed by the top tier user */
        public String[] topTierRoutes() {...}
    }
    ...
}
```

Código 2.1: Rutas utilizadas para simplificar el proceso de autorización de los usuarios.

2.1.3. STRATEGY

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetur id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio

metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetur.

Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas

dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.

Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus.

Bibliografía

- [1] V. Sarcar, *Java Design Patterns: A Hands-On Experience with Real-World Examples*. Apress, 2018.