TALLER 5

Link: https://github.com/percyvega/legacy-adapter.git

Información general del proyecto: para qué sirve, cuál es la estructura general del diseño, qué grandes retos de diseño enfrenta (i.e. ¿qué es lo difícil?). Deben incluir la URL para consultar el proyecto.

Este código modela entidades en java que están relacionadas con la gestión de clientes corporativos en MotionPoint.

tiene una interfaz Company (abstrae los detalles de la información del cliente) que es implementada por CompanyLmpl, que maneja la lógica de validación y almacenamiento

tiene una interfaz website(abstrae los dettalles de la información del sitio web

) que es implementada por WbsiteLmpl, que maneja la lógica de validación y almacenamiento

tiene una clase abstracta que se llama CompanyUtil, da funciones de utilidad

Lo difícil, del proyecto a mi forma de verlo es como se implementan dos interfaces la del cliente y la del website, o la validación del url. Además, la abstracción de datos para representar la información de manera adecuada

• Información y estructura del fragmento del proyecto donde aparece el patrón. No se limite únicamente a los elementos que hacen parte del patrón: para que tenga sentido su uso, probablemente va a tener que incluir elementos cercanos que sirvan para contextualizarlo.

Se usa patrones como la Abstracción Factory a la hora de proporcionar las interfaces pare representar a las empresas y sitio web. En Company y Website

Se usa Constructor para proporcionar ciertos valores durante la creación del objeto. En CompanyImpl y WebsiteImpl

Se usa Facade para dar métodos estáticos de imprimir información sobre compañía y sitio web, similar porque proporciona una interfaz simple para funcionalidades mas grandes

• Información general sobre el patrón: qué patrón es y para qué se usa usualmente.

Abstracción Factory: Un patrón que proporciona una interfaz para crear familias de objetos relacionados o dependientes sin especificar sus clases concretas.

Patron Builder: Un patrón que separa la construcción de un objeto complejo de su representación, permitiendo que el mismo proceso de construcción cree varias representaciones.

Patron Facade: Un patrón que proporciona una interfaz simplificada a un conjunto de interfaces en un subsistema, facilitando su uso.

• Información del patrón aplicado al proyecto: explicar cómo se está utilizando el patrón dentro del proyecto.

Abstracción: Company y Website, son clases que definen métodos y presentan la estructura esencial de la compañía y sitio web, mientras que CompanyImpl y websiteImpl, implentan estas interfaces con la implementación correcta de las abstracciones

Builder: no se usa específicamente un builder pero CompanyImpl y websiteImpl puede verse como una forma simple de crear objetos

Facade: Company útil, porque utiliza métodos estáticos que pueden funcionar como una interfaz mas simple, para imprimir información. printwebsite, oculta detalles de implementación, para proporcionar una información mas sencilla

• ¿Por qué tiene sentido haber utilizado el patrón en ese punto del proyecto? ¿Qué ventajas tiene?

Abstraction is a key concept in programming, as it allows you to create more modular and reusable code. Using abstractions hides implementation details and exposes only interfaces that define the expected behavior. This makes code maintenance easier, since changes to an abstraction do not affect other components that use it.

El patrón de diseño facade permite simplificar el acceso a una funcionalidad compleja mediante una interfaz más simple y amigable. Así, los clientes que usan el facade no tienen que lidiar con los detalles internos de la funcionalidad, sino que pueden enfocarse en sus propias necesidades. Este patrón es útil para mejorar la legibilidad y la mantenibilidad del código, especialmente en contextos estudiantiles donde se busca aprender los conceptos básicos de la programación orientada a objetos.

Una de las ventajas de usar el patrón Builder es que el código para la creación de objetos es más legible, y también a la hora de la construcción de builder o company el builder desacopla este proceso de la clase principal. Esto permite que el código sea más modular y fácil de mantener.

• ¿Qué desventajas tiene haber utilizado el patrón en ese punto del proyecto?

Abstracción: puede agregarle complejidad al código y su comprensión, también aumenta la cantidad de clases, mayor dificultad de comprensión

Builder: Puede parecer muy compleja cuando se utiliza en objetos simples.

Facade: Esta bien acoplada, esto hace que un cambio en las otras clases puede afectar la interfaz o viceversa.

• ¿De qué otras formas se le ocurre que se podrían haber solucionado, en este caso particular, los problemas que resuelve el patrón?

Abstracción: Se puede hacer una herencia simple, o trabajar directamente con las clases concretas, tipo usar company en vez de CompanyImpl

Builder: Utilizar unos métodos de fabricación para crear instancia con configuraciones especificas

Facade: Hacer que las clases tuvieran una interacción directa con los subsistemas