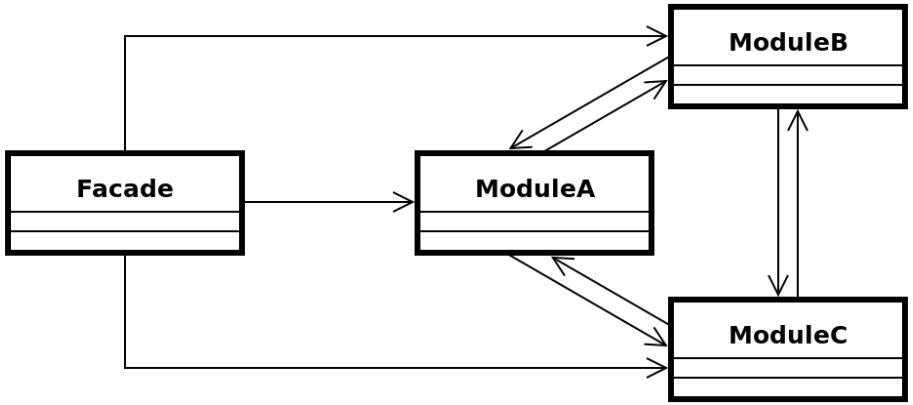


Patron de diseño “Fachada”

Raul Aleandro Martinez Jaramillo

Intension

- Proporcionar una interfaz unificada para un conjunto de interfaces en un subsistema. Facade define una interfaz de nivel superior que hace que el subsistema sea más fácil de usar.
- Envuelva un subsistema complicado con una interfaz más simple.

Problema

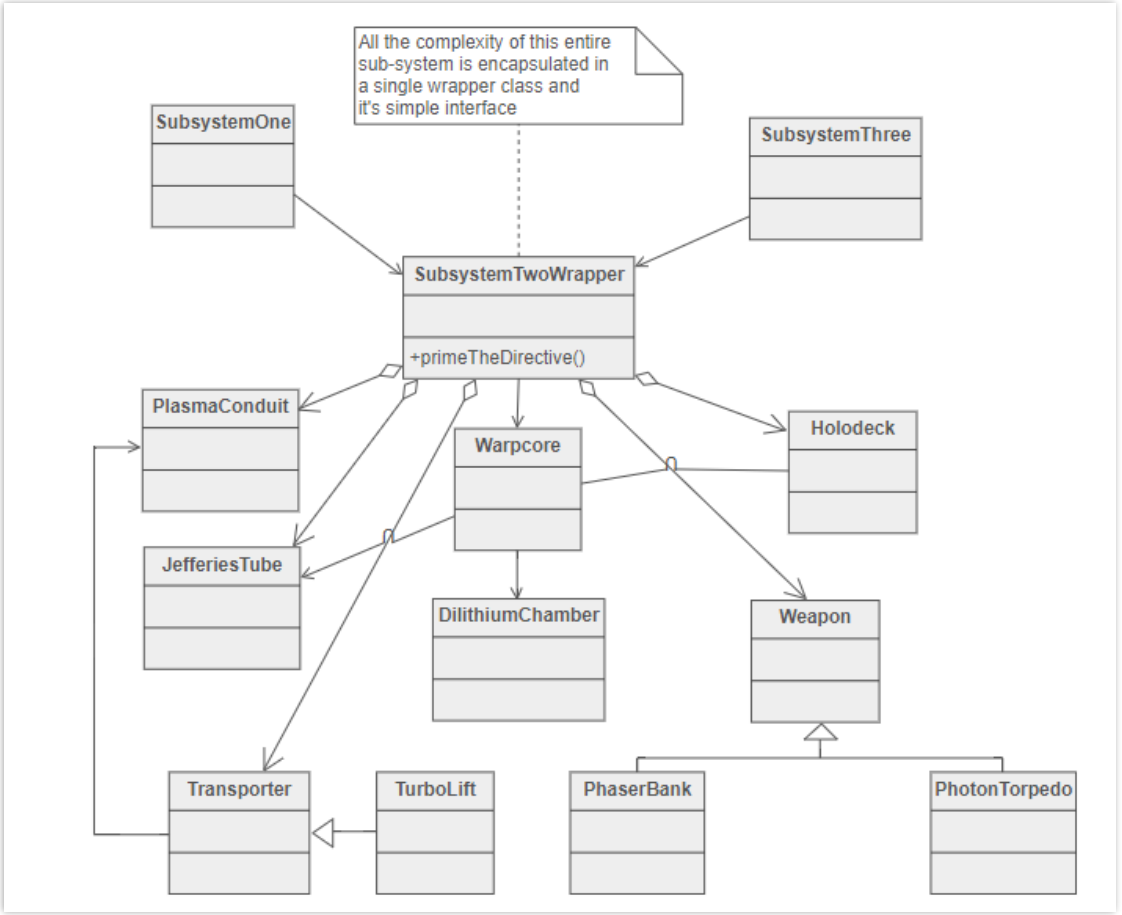
- Un segmento de la comunidad cliente necesita una interfaz simplificada para la funcionalidad general de un subsistema complejo.

Discusión

- Facade discute la encapsulación de un subsistema complejo dentro de un único objeto de interfaz. Esto reduce la curva de aprendizaje necesaria para aprovechar con éxito el subsistema. También promueve la disociación del subsistema de sus potenciales clientes. Por otro lado, si Fachada es el único punto de acceso para el subsistema, limitará las características y la flexibilidad que puedan necesitar los "usuarios avanzados".

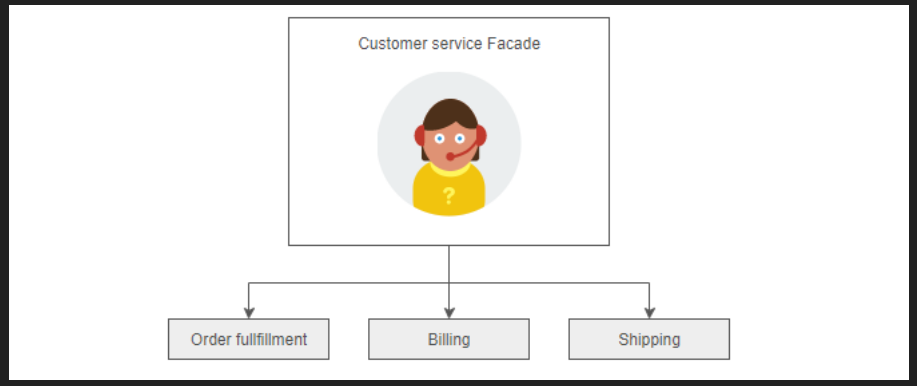
- El objeto Fachada debe ser un defensor o facilitador bastante simple. No debe convertirse en un oráculo omnisciente o un objeto "dios".

Estructura


All the complexity of this entire sub-system is encapsulated in a single wrapper class and it's simple interface
SubsystemOne
SubsystemThree
SubsystemTwoWrapper
+primeTheDirective()
PlasmaConduit
Warpcore
Holodeck
JefferiesTube
DilithiumChamber
Weapon
Transporter
TurboLift
PhaserBank
PhotonTorpedo

Ejemplo

- Facade define una interfaz unificada de nivel superior para un subsistema que hace que sea más fácil de usar. Los consumidores encuentran una Fachada cuando ordenan desde un catálogo. El consumidor llama a un número y habla con un representante de servicio al cliente. El representante de servicio al cliente actúa como una fachada, proporcionando una interfaz para el departamento de cumplimiento de pedidos, el departamento de facturación y el departamento de envío.



Lista de Chequeo

- Identifique una interfaz unificada más simple para el subsistema o componente.
- Diseña una clase 'envoltura' que encapsula el subsistema.
- La fachada / envoltorio captura la complejidad y las colaboraciones del componente y delega los métodos apropiados.
- El cliente usa (está acoplado a) la Fachada solamente.
- Considere si Fachadas adicionales agregarían valor.

Reglas Generales

- Facade define una nueva interfaz, mientras que Adapter usa una interfaz antigua. Recuerde que Adapter hace que dos interfaces existentes funcionen juntas en lugar de definir una interfaz completamente nueva.
- Abstract Factory se puede utilizar como una alternativa a Facade para ocultar clases específicas de la plataforma.
- Los objetos de fachada a menudo son Singleton porque solo se requiere un objeto Fachada.
- Pregunta: ¿Entonces la diferencia entre el patrón de Adaptador y el patrón de Fachada es que el Adaptador envuelve una clase y la Fachada puede representar muchas clases?
- Respuesta: ¡No! Recuerde, el patrón Adapter cambia la interfaz de una o más clases en una interfaz que espera un cliente.