Actividad 5

Pablo Sanchez Galdamez (21001135)

Contenidos

5 Patrones de Diseño

MVC

Definición

Modelo Vista Controlador (MVC) es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

Características

Este es un modelo muy maduro y que ha demostrado su validez a lo largo de los años en todo tipo de aplicaciones, y sobre multitud de lenguajes y plataformas de desarrollo. Este se separa en:

- **El Modelo:** que contiene una representación de los datos que maneja el sistema, su lógica de negocio, y sus mecanismos de persistencia
- **La Vista:** o interfaz de usuario, que compone la información que se envía al cliente y los mecanismos interacción con éste
- **El Controlador:** que actúa como intermediario entre el Modelo y la Vista, gestionando el flujo de información entre ellos y las transformaciones para adaptar los datos a las necesidades de cada uno

Ventajas

- La implementación se realiza de forma modular
- Las modificaciones a las vistas no afectan al modelo de dominio, simplemente se modifica la representación de la información, no su tratamiento
- MVC esta demostrando ser un patrón de diseño bien elaborado pues las aplicaciones que lo implementan presentan una extensibilidad y una mantenibilidad únicas comparadas con otras aplicaciones basadas en otros patrones

Desventajas

- Para desarrollar una aplicación bajo el patrón de diseño MVC es necesario una mayor dedicación en los tiempos iniciales del desarrollo. Normalmente

el patrón exige al programador desarrollar un mayor número de clases que, en otros entornos de desarrollo, no son necesarias.
- MVC requiere la existencia de una arquitectura inicial sobre la que se deben construir clases e interfaces para modificar y comunicar los módulos de una aplicación. Esta arquitectura inicial debe incluir, por lo menos, un mecanismo de eventos para poder proporcionar las notificaciones que genera el modelo de aplicación; una clase Modelo, otra clase Vista y una clase Controlador genéricas que realicen todas las tareas de comunicación, notificación y actualización que serán luego transparentes para el desarrollo de la aplicación.
- MVC es un patrón de diseño orientado a objetos por lo que su implementación es sumamente costosa y difícil en lenguajes que no siguen este paradigma.

Singleton

Definición

Es un patrón de diseño que permite restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto.

Características

- La propia clase es responsable de crear la única instancia. Por medio de su método constructor.
- Permite el acceso global a dicha instancia mediante un método de clase.
- Declara el constructor de clase como privado para que no sea instanciable directamente.
- Al estar internamente autoreferenciada, en lenguajes como Java, el recolector de basura no actúa

Ventajas

Este patrón es perfecto para aquellas circunstancias en las que la clase controla el acceso a un recurso físico único, o para cierto tipo de datos que tiene que estar disponible para toda la aplicación.

Desventajas

- Incluye restricciones en su uso
- Requiere de gestionar un estado global
- Es complejo de implementar en un aplicación que necesite correr en múltiples hilos.

Factory

Definición

Es un enfoque de programación que sirve para crear objetos sin tener que especificar su clase exacta. Esto quiere decir que el objeto creado puede intercambiarse con flexibilidad y facilidad. Para implementar este método, los desarrolladores utilizan el Factory Method, que da nombre a este patrón.

Características

Suele estar formada por los siguientes elementos:

- Cliente: Quien necesita crear los objetos
- Fabrica Abstracta: Clase encargada de crear los objetos
- Fabrica Concreta: Es la que provee la instancia concreta que se desea
- Producto Abstracto: Es la definición genérica
- Producto Concreto: Es el objeto final que sera utilizado

Ventajas

- Brinda flexibilidad al aislar a los productos concretos.
- Facilita cambiar familias de productos

Desventajas

- Para agregar productos nuevos se tienen que modificar todas las fabricas
- Añade un nivel de complejidad que puede ser excesivo en aplicaciones pequeñas

Adapter

Definición

Adapter es un patrón de diseño estructural que permite la colaboración entre objetos con interfaces incompatibles.

Características

El funcionamiento se da de la siguiente manera:

1. El adaptador obtiene una interfaz compatible con uno de los objetos existentes.
2. Utilizando esta interfaz, el objeto existente puede invocar con seguridad los métodos del adaptador.
3. Al recibir una llamada, el adaptador pasa la solicitud al segundo objeto, pero en un formato y orden que ese segundo objeto espera.

Ventajas

- Puedes separar la interfaz o el código de conversión de datos de la lógica de negocio primaria del programa.
- Puedes introducir nuevos tipos de adaptadores al programa sin descomponer el código cliente existente, siempre y cuando trabajen con los adaptadores a través de la interfaz con el cliente.

Desventajas

La complejidad general del código aumenta, ya que debes introducir un grupo de nuevas interfaces y clases. En ocasiones resulta más sencillo cambiar la clase de servicio de modo que coincida con el resto de tu código.

Observer

Definición

Observer es un patrón de diseño de comportamiento que te permite definir un mecanismo de suscripción para notificar a varios objetos sobre cualquier evento que le suceda al objeto que están observando.

Características

El funcionamiento se da de la siguiente manera:

1. El Notificador envía eventos de interés a otros objetos.
2. Cuando sucede un nuevo evento, el notificador recorre la lista de suscripción e invoca el método de notificación declarado en la interfaz suscriptora en cada objeto suscriptor.
3. La interfaz Suscriptora declara la interfaz de notificación.
4. Los Suscriptores Concretos realizan algunas acciones en respuesta a las notificaciones emitidas por el notificador.
5. El Cliente crea objetos tipo notificador y suscriptor por separado y después registra a los suscriptores para las actualizaciones del notificador.

Ventajas

- Puedes introducir nuevas clases suscriptoras sin tener que cambiar el código de la notificadora (y viceversa si hay una interfaz notificadora).
- Puedes establecer relaciones entre objetos durante el tiempo de ejecución.

Desventajas

- Los suscriptores son notificados en un orden aleatorio.

Cuadro Comparativo

.	Ventajas	Desventajas
MVC	Este se implementa de forma modular, y ya a quedado demostrado con el tiempo que es un patrón solido para utilizar	Requiere de un framework que defina las interfaces, y de un costo mayor al inicio del desarrollo
Singleton	Controla perfectamente un estado global o un recurso único	Dificulta el trabajar con varios hilos, y por la naturaleza de manejar un estado global es difícil de testear
Factory	Brinda flexibilidad y facilita cambiar entre objetos	Añade una complejidad innecesaria en aplicaciones pequeñas
Adapter	Separa la lógica de la conversión de datos, y se puede extender fácilmente	Requiere de crear un grupo completamente nuevo de interfaces y clases cuando puede ser menos costoso modificar el código que ya se tiene
Observer	Se puede extender fácilmente y relaciona de forma clara los objetos.	No se tiene un buen control sobre el orden en el que se notifica a los subscriptores