Prácticas GoF

1. Prácticas patrones creacionales
   1. Tenemos un Servicio de autenticación que recibe un objeto de tipo Usuario, el cual contiene un nombre y una clave, y su único método devuelve un valor booleano: cierto, usuario autenticado, falso, usuario no autenticado. Implementar mediante el patrón de instancia única (singleton)
   2. Tenemos un sistema que devuelve empleados fijos o empleados a tiempo parcial nuevos según se le solicite, de modo que dicho sistema pueda rellenar el objeto devuelto con datos. Implementar mediante el patrón método de factoría (Factory method)
2. Prácticas patrones estructurales
   1. Una aplicación web recibe desde una página html una petición GET acabada en “xyz”, esto es, un nombre lógico. Por ejemplo “cliente.xyz”. Una servlet analiza la petición y devuelve en respuesta una página JSP física. Implementar mediante un patrón modelo, vista, controlador (MVC)
   2. Tenemos un cliente que autentica al acceso a sus sistemas empleando instancias de la clase Empleado, en la cual, entre otros valores, hay un nombre de usuario y una clave. Queremos reutilizar nuestro Servicio de Autenticación sin modificar su código. Resolverlo mediante un patrón adaptador (Object Adapter)
   3. Dado un sistema gráfico existente capaz de dibujar componentes, queremos tener un botón cuyo texto aparezca siempre justificado a la derecha, pero no deseamos cambiar el código de la clase Botón existente, ni emplear la herencia. Implementar mediante un patrón estético (Decorator)
3. Prácticas patrones de comportamiento
   1. Queremos validar los objetos de tipo Empleado de varias formas complejas en secuencia, y no queremos introducir todo el código necesario en una sola clase. El sistema de validación debe lanzar una excepción en cuanto una validación falle. Implementar mediante una cadena de responsabilidad (Chain of Responsability)
   2. Tenemos un sistema de gestión de archivos que sabe copiar un archivo, borrarlo y decirnos si es un directorio. El sistema espera recibir las siguientes órdenes por consola:
      1. copiar nombre\_archivo nuevo\_nombre
      2. borrar nombre\_archivo
      3. esdirectorio nombre\_archivo

Implementar mediante un patrón orden (command)

* 1. En el sistema anterior, cada vez que se ejecuta una orden se genera un evento que incluye información de contexto. Queremos poder recoger ese evento y procesarlo de diversas formas, contemplando la posibilidad de necesidades futuras, y sin que dichas necesidades afecten al código existente. Implementar mediante un patrón Observador (Observer)