BIG BANG
programa de desarrollo de Crux Consultores

Patrones de Diseño

Agenda

- Fundamentos
- Tipos
- Ejemplos

BIG BANG
programa de desarrollo de Crux Consultores

Fundamentos

Patrones de Diseño

- Los patrones de diseño son el esqueleto de las soluciones a problemas comunes en el desarrollo de software.

- Brindan una solución ya probada y documentada a problemas de desarrollo de software que están sujetos a contextos similares.

Elementos de un patrón de diseño

- Nombre.
- Problema: Cuando aplicar el patrón.
- La solución: descripción del patrón.
- Consecuencias: costos y beneficios.

Tipos de Patrones

Patrones Creacionales

- Utilizados para la inicialización y configuración de objetos.

Ejemplos: Patrones Creacionales

Fábrica Abstracta (Abstract Factory)

- El problema a solucionar por este patrón es el de crear diferentes familias de objetos, como por ejemplo la creación de interfaces gráficas de distintos tipos (ventana, menú, botón, etc.).

Ejemplos: Patrones Creacionales

Fábrica Abstracta (Factory Method)

- Parte del principio de que las subclases determinan la clase a implementar.

```
public class ConcreteCreator extends Creator
  {
  protected Product FactoryMethod()
      {
            return new ConcreteProduct();
      }
}
public interface Product{}
public class ConcreteProduct implements Product{}
      public class Client
      {
            public static void main(String args[])
            {
                  Creator UnCreator;
                  UnCreator = new ConcreteCreator();
                  UnCreator.AnOperations();
            }
      }
```

Ejemplos: Patrones Creacionales

Singleton: Restringe la instanciación de una clase o valor de un tipo a un solo objeto.

```
public sealed class Singleton
{
      private static volatile Singleton instance;
      private static object syncRoot = new Object();
      private Singleton()
      {
            System.Windows.Forms.MessageBox.Show("Nuevo Singleton");
      }
      public static Singleton GetInstance
      {
            get
            {
                  if (instance == null)
                  {
                        lock(syncRoot)
                        {
                              if (instance == null)
                                    instance = new Singleton();
                        }
                  }
                  return instance;
            }
      }
}
```

Ejemplos: Patrones Creacionales

MVC (Model View Controller)

- Este patrón plantea la separación del problema en tres capas:
  - La capa model, que representa la realidad.
  - La capa controller , que conoce los métodos y atributos del modelo, recibe y realiza lo que el usuario quiere hacer.
  - La capa vista, que muestra un aspecto del modelo y es utilizada por la capa anterior para interaccionar con el usuario.

Patrones Estructurales

- Separan la interfaz de la implementación. Se ocupan de cómo las clases y objetos se agrupan, para formar estructuras más grandes.

Ejemplos: Patrones Estructurales

- Adaptador (*Adapter*): Convierte una interfaz en otra.
- Puente (*Bridge*): Desacopla una abstracción de su implementación permitiendo modificarlas independientemente.
- Objeto Compuesto (*Composite*): Utilizado para construir objetos complejos a partir de otros más simples, utilizando para ello la composición recursiva y una estructura de árbol.

Ejemplos: Patrones Estructurales

- Envoltorio (*Decorator*): Permite añadir dinámicamente funcionalidad a una clase existente, evitando heredar sucesivas clases para incorporar la nueva funcionalidad.

- Fachada (*Facade*): Permite simplificar la interfaz para un subsistema.

- Peso Ligero (*Flyweight*): Elimina la redundancia o la reduce cuando tenemos gran cantidad de objetos con información idéntica.

Patrones de Comportamiento

- Más que describir objetos o clases, describen la comunicación entre ellos.

Ejemplos: Patrones de Comportamiento

- Cadena de responsabilidad (*Chain of responsibility*): La base es permitir que más de un objeto tenga la posibilidad de atender una petición.

- Iterador (*Iterator*): Define una interfaz que declara los métodos necesarios para acceder secuencialmente a una colección de objetos sin exponer su estructura interna.

Ejemplos: Patrones de Comportamiento

- Mediador (*Mediator*): Coordina las relaciones entre sus asociados. Permite la interacción de varios objetos, sin generar acoples fuertes en esas relaciones.

- Recuerdo (*Memento*): Almacena el estado de un objeto y lo restaura posteriormente.

- Estado (*Server*): Se utiliza cuando el comportamiento de un objeto cambia dependiendo del estado del mismo.

Ejemplos: Patrones de Comportamiento

- Observador (*Observer*): Notificaciones de cambios de estado de un objeto.

```
Public Class Articulo
   Delegate Sub DelegadoCambiaPrecio(ByVal unPrecio As Object)
   Public Event CambiaPrecio As DelegadoCambiaPrecio
   Dim _cambiaPrecio As Object
   Public WriteOnly Property Precio()
      Set(ByVal value As Object)
         _cambiaPrecio = value
         RaiseEvent CambiaPrecio(_cambiaPrecio)
      End Set
   End Property
End Class
Public Class ArticuloObservador
   Public Sub Notify(ByVal unObjecto As Object)
      Console.WriteLine("El nuevo precio es:" & unObjecto)
   End Sub
```

BIG BANG
programa de desarrollo de Crux Consultores

Patrones de Diseño