Gang of Four (GoF) Design Patterns - Interview Guide

This document provides concise explanations of all 23 GoF design patterns, with examples relevant to .NET development, to assist in technical interview preparation.

Creational Patterns

- Singleton

Ensures a class has only one instance and provides a global point of access.

Example: DbContext in Entity Framework.

- Factory Method

Defines an interface for creating objects but lets subclasses alter the type of objects created.

Example: LoggerFactory in ASP.NET Core.

- Abstract Factory

Provides an interface to create families of related or dependent objects without specifying their concrete classes.

Example: UI component factories for cross-platform apps.

- Builder

Separates the construction of a complex object from its representation.

Example: StringBuilder for constructing strings efficiently.

- Prototype

Creates new objects by copying an existing object (prototype).

Example: ICloneable interface in .NET.

Structural Patterns

- Adapter

Converts the interface of a class into another interface the client expects.

Example: IEnumerable to IQueryable conversion in LINQ.

- Bridge

Decouples abstraction from implementation, allowing them to vary independently.

Example: Logging frameworks in .NET.

- Composite

Treats individual objects and compositions of objects uniformly.

Example: WPF visual tree (UIElement, Panel, etc.).

- Decorator

Adds responsibilities to an object dynamically without altering its structure.

Example: Streams in .NET like FileStream and BufferedStream.

- Facade

Provides a unified interface to a set of interfaces in a subsystem.

Example: Startup.cs in ASP.NET Core for configuring services and middleware.

- Flyweight

Minimizes memory use by sharing as much data as possible with similar objects.

Example: String interning in .NET.

- Proxy

Provides a placeholder or surrogate to control access to another object.

Example: WCF Service Proxies.

Behavioral Patterns

- Chain of Responsibility

Passes requests along a chain of handlers until one handles it.

Example: ASP.NET Core middleware pipeline.

- Command

Encapsulates a request as an object, allowing parameterization and queuing.

Example: RelayCommand in WPF MVVM pattern.

- Interpreter

Defines a representation for a language's grammar and interprets sentences in the language.

Example: Expression trees in LINQ.

- Iterator

Provides a way to access elements of a collection sequentially without exposing its underlying representation.

Example: IEnumerable and IEnumerator in .NET.

- Mediator

Encapsulates how a set of objects interact.

Example: Mediator pattern in MediatR library for CQRS.

- Memento

Captures and restores an object's internal state without violating encapsulation.

Example: Undo functionality in text editors.

- Observer

Defines a dependency between objects so that when one changes state, all dependents are notified.

Example: INotifyPropertyChanged in WPF/WinForms.

- State

Allows an object to alter its behavior when its internal state changes.

Example: State pattern in workflow engines.

- Strategy

Defines a family of algorithms, encapsulates each one, and makes them interchangeable.

Example: Sorting algorithms in LINQ's OrderBy.

- Template Method

Defines the skeleton of an algorithm and lets subclasses redefine certain steps.

Example: DbContext.SaveChanges in EF Core.

- Visitor

Separates an algorithm from the object structure it operates on.

Example: Expression Visitors in LINQ.