

# Patrones de Diseño en PHP

*Módulo: Desarrollo Web en Entorno Servidor (DAW)*

---

## 1. Introducción

Los patrones de diseño son soluciones reutilizables y probadas para problemas comunes en el diseño de software orientado a objetos.

Su conocimiento y correcta aplicación permiten desarrollar código más limpio, mantenible y escalable.

En esta actividad trabajaréis en **grupos de 2 alumnos** para investigar, comprender y exponer uno de los patrones de diseño del catálogo referenciado. Cada grupo debe escoger un patrón distinto.

La duración total del ejercicio, incluyendo exposición, será de **máximo 4 horas**.

---

## 2. Objetivos del ejercicio

- Comprender el propósito y uso de los patrones de diseño.
  - Identificar cuándo aplicar un patrón específico en un proyecto PHP.
  - Desarrollar una mini-aplicación de ejemplo que implemente el patrón elegido.
  - Mejorar las habilidades de presentación y trabajo en equipo.
- 

## 3. Desarrollo del trabajo

1. Accedéis al catálogo de patrones en la web del recurso:  
[Catálogo de Patrones de Diseño – Refactoring Guru](#)
2. Cada grupo elige un patrón de diseño (ningún grupo puede repetir el mismo patrón).
3. Investigáis y preparáis una presentación que incluya:
  - Definición del patrón y el problema que resuelve.
  - Estructura (diagrama UML o pseudo-UML) y componentes principales.
  - Ejemplo de uso práctico en **PHP**.
  - Ventajas y desventajas del patrón.

4. Desarrolláis una mini-aplicación en PHP que demuestre el funcionamiento del patrón elegido (por ejemplo: módulo sencillo, script o prototipo).
  5. Preparáis una exposición (máximo de 10 minutos) que puede combinar diapositivas + demo en vivo.
- 

#### 4. Criterios de evaluación

- Claridad en la explicación teórica del patrón (25 %).
  - Calidad técnica del ejemplo en PHP (25 %).
  - Creatividad y diseño de la exposición (20 %).
  - Trabajo en equipo y reparto de tareas (15 %).
  - Cumplimiento del tiempo y originalidad (15 %).
- 

#### 5. Consejos del profesor 💡

Elige un patrón que realmente te interese o que creas que puede ser útil para tus futuros proyectos en PHP. La comprensión profunda de su estructura y propósito es más importante que memorizar la teoría.

Piensa en un ejemplo sencillo pero ilustrativo (por ejemplo: sistema de gestión, módulo de logging, generador de informes, etc.).

Reparte tareas claramente entre los dos miembros del grupo: investigación, código, presentación, demo.

---

#### 6. Entrega

Cada grupo entregará al final de la sesión:




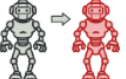

- Código fuente de la mini-aplicación (ficheros PHP + cualquier recurso necesario).
- Presentación (diapositivas u otro formato) o demo en vivo.
- Un archivo breve (máx. 1 página) con el resumen del patrón: definición, cuándo usarlo, ventajas e inconvenientes.
- Se realizará la exposición delante del resto de la clase.

## 7. Catálogo de Patrones del recurso

A continuación tenéis la clasificación según el catálogo del recurso: [refactoring.guru](https://refactoring.guru)

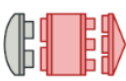


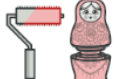

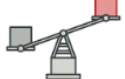
### Patrones creacionales

Estos patrones proporcionan mecanismos de creación de objetos que incrementan la flexibilidad y la reutilización del código existente.

 Factory Method	 Abstract Factory
 Builder	 Prototype
 Singleton	





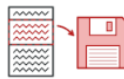

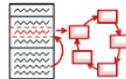
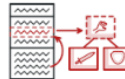


### Patrones estructurales

Estos patrones explican cómo ensamblar objetos y clases en estructuras más grandes, mientras se mantiene la flexibilidad y eficiencia de la estructura.

 Adapter	 Bridge
 Composite	 Decorator
 Facade	 Flyweight

### Patrones de comportamiento

Estos patrones tratan con algoritmos y la asignación de responsabilidades entre objetos.

 Chain of Responsibility	 Command	 Iterator	 Mediator
 Memento	 Observer	 State	 Strategy
 Template Method	 Visitor		