**Resumo – Padrões GoF de Criação**

**Objetivo:**  
Aprender como criar objetos de forma flexível, reduzindo o acoplamento entre módulos em sistemas orientados a objetos, usando cinco padrões de criação:  
**Factory Method, Abstract Factory, Builder, Prototype e Singleton**.

**Importância:**  
Evitar dependências diretas entre classes concretas, seguindo princípios SOLID (especialmente Open-Closed e Dependency Inversion), aumentando a flexibilidade e facilitando a evolução dos sistemas.

**Padrões estudados:**

**1. Factory Method**

* **Problema: Criar objetos diretamente gera dependência entre classes.**
* **Solução: Definir um método de fábrica abstrato e delegar às subclasses a responsabilidade de instanciar o objeto concreto.**
* **Exemplo: iterator() no Java Collections**

**2. Abstract Factory**

* **Problema: Criar famílias de objetos relacionados sem acoplamento direto.**
* **S Def**
* **Exemplo: Decodificação de mensagens em XML, CSV ou texto fixo.**

**3. Builder**

* **Problema: Construir objetos complexos com diferentes representações.**
* **Solução: Separar a construção do objeto da sua representação final, permitindo reutilizar o processo para diferentes formatos.**
* **Exemplo: Exportação de notas financeiras em PDF, XML ou XLS.**

**4.**

* **Problema: Criar muitas variações de objetos a partir de uma estrutura comum.**
* **Solução: Criar novos objetos a partir da clonagem de protótipos previamente registrados.**
* **Exemplo: Mensagens diferentes para clientes conforme a origem.**

**5. Singleton**

* **Problema: Garantir que apenas uma instância de uma classe exista em todo o sistema.**
* **Solução: Controlar a criação e fornecer um ponto de acesso global para essa instância.**
* **Observação: Uso deve ser cuidadoso, pois pode dificultar testes e flexibilidades futuras.**

**Conclusão:**

* O uso adequado dos padrões de criação melhora a manutenção, evolução e flexibilidade dos sistemas.
* Cada padrão resolve problemas específicos de criação de objetos.
* Singleton deve ser usado com cuidado para não comprometer a qualidade do software.

