**Defesa e Justificativa de Padrão de projeto adequado**

Para o cenário descrito, onde uma empresa precisa criar um sistema para uma academia de musculação/crossfit que apresenta indicações de exercícios divididos em categorias (corrida e musculação) com subcategorias (resistência ou velocidade), o padrão de projeto mais adequado seria o **Abstract Factory**.

O **Abstract Factory** é um padrão de projeto de criação que fornece uma interface para criar famílias de objetos relacionados ou dependentes sem especificar suas classes concretas. Nesse caso, as categorias (corrida e musculação) e suas subcategorias (resistência ou velocidade) podem ser tratadas como famílias de objetos.Descrevo abaixo os motivos pelos quais se faz necessario escolher este padrão de projeto:

**1.Variedade de Categorias e Subcategorias:** O padrão Abstract Factory é adequado quando há uma variedade de famílias de objetos a serem criadas. Nesse caso, temos as categorias (corrida e musculação) que têm subcategorias (resistência ou velocidade). O Abstract Factory permite criar objetos de diferentes categorias e subcategorias sem depender das classes concretas, proporcionando flexibilidade.

**2. Separação de Responsabilidades:** O padrão Abstract Factory ajuda a separar a criação de objetos da sua utilização. Isso é benéfico para um sistema de academia, pois a lógica para selecionar exercícios, categorias e subcategorias pode ser isolada da lógica de criação de objetos.

**3.Extensibilidade**: O sistema de academia pode evoluir com o tempo, adicionando mais categorias, subcategorias e exercícios. O Abstract Factory permite adicionar novas famílias de objetos (novas categorias, subcategorias) facilmente, pois novas fábricas concretas podem ser introduzidas sem modificar o código existente.

**4. Consistência:** O padrão Abstract Factory garante que objetos criados estejam em conformidade com uma família específica. Isso é útil para manter a consistência nas indicações de exercícios, garantindo que os exercícios selecionados pertençam às categorias e subcategorias corretas.

**5. Isolamento de Mudanças Futuras:** Se houver alterações nas classes de exercícios, categorias ou subcategorias, o restante do código não será afetado, desde que a interface da fábrica abstrata seja mantida constante.

Portanto, o padrão Abstract Factory é a escolha mais apropriada para o projeto da academia de musculação/crossfit, pois oferece uma estrutura que permite criar famílias de objetos (categorias, subcategorias) de forma flexível, extensível e bem organizada.

Neste exemplo que envio juntamente em anexo, criei no neatbeans interfaces e classes concretas para representar os diferentes tipos de exercícios (corrida e musculação) e suas subcategorias (resistência e velocidade). As fábricas concretas implementam a interface de fábrica abstrata e são usadas para criar os exercícios específicos. O método main do AcademiaApp ilustra como criar um exercício de corrida de resistência usando a fábrica apropriada.

Barbara Machado Faria Neves