**1. Factory Method**

**Resumo:**  
O padrão Factory Method define uma interface para a criação de objetos, mas delega às subclasses a decisão sobre qual classe concreta instanciar. Assim, o processo de criação é centralizado em um método que pode ser sobrescrito para produzir diferentes tipos de objetos. (não expõe sua implementação por conta de deixar as subclasses fazerem isso)

**Vantagens:**

* Promove o desacoplamento entre o código cliente e as classes concretas.
* Facilita a extensão e a manutenção, já que novas classes podem ser adicionadas sem alterar o código existente.

**Desvantagens:**

* Pode aumentar a complexidade do sistema por exigir a criação de subclasses adicionais para cada variação do produto.

Diagrama

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.

**2. Abstract Factory**

**Resumo:**  
O Abstract Factory oferece uma interface para criar famílias de objetos relacionados ou dependentes sem especificar suas classes concretas. Esse padrão é útil quando se deseja garantir que os objetos criados sejam compatíveis entre si.

(O padrão Abstract Factory pode envolver a aplicação do padrão Factory Method, pois ambos os padrões são utilizados para criar objetos de forma abstrata.)

**Vantagens:**

* Garante a consistência entre os produtos, pois as famílias de objetos são criadas conjuntamente.
* Facilita a troca de famílias de produtos sem modificar o código cliente.

**Desvantagens:**

* Sua implementação pode ser complexa e aumentar o número de classes no sistema.
* Adicionar novas famílias pode exigir mudanças abrangentes no código.

Diagrama

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

O conteúdo gerado por IA pode estar incorreto.

**3. Builder**

**Resumo:**  
O padrão Builder separa a construção de um objeto complexo da sua representação, permitindo que o mesmo processo de construção possa criar diferentes representações do objeto. Ele concentra-se no passo a passo da criação, isolando as etapas de montagem.

**Vantagens:**

* Permite a criação de objetos complexos de forma incremental e controlada.
* Facilita a criação de diferentes representações a partir do mesmo processo de construção.

**Desvantagens:**

* Pode resultar em uma quantidade maior de classes e aumentar a complexidade do código.
* Exige um planejamento cuidadoso para garantir que todas as etapas da construção sejam contempladas.

Diagrama

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, chat ou mensagem de texto

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.

**4. Prototype**

**Resumo:**  
O padrão Prototype cria objetos clonando uma instância prototípica. Em vez de instanciar diretamente classes concretas, utiliza a clonagem para copiar os atributos e comportamentos já definidos.

(Reduzir a necessidade de criar hierarquias paralelas de classes ou de criar subclasses apenas para instanciar objetos específicos é um dos efeitos obtidos com a aplicação)

**Vantagens:**

* Pode melhorar o desempenho quando a criação de um objeto a partir do zero é custosa.
* Facilita a criação de cópias de objetos complexos.

**Desvantagens:**

* A implementação do método de clonagem pode ser complexa, especialmente em relação à distinção entre cópias rasas e profundas.
* Pode causar problemas se os objetos clonados mantiverem referências indesejadas ou estados compartilhados.

Diagrama

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.

**5. Singleton**

**Resumo:**  
O padrão Singleton garante que uma classe tenha somente uma instância e fornece um ponto global de acesso a essa instância. É utilizado quando é necessário controlar o acesso a recursos compartilhados.

(utiliza um construtor private e implementar uma operação estática que retorna uma instância criada em uma operação com escopo static e armazenada como um atributo static dentro da própria classe)

**Vantagens:**

* Assegura o controle centralizado de acesso a recursos, evitando a criação de múltiplas instâncias.
* Simplifica a coordenação de ações em sistemas onde um único ponto de acesso é desejado.

**Desvantagens:**

* Pode introduzir um estado global, dificultando testes unitários e a manutenção do código.
* Se não for implementado corretamente, especialmente em ambientes concorrentes, pode causar problemas de sincronização e desempenho.

Diagrama

O conteúdo gerado por IA pode estar incorreto.

Texto

O conteúdo gerado por IA pode estar incorreto.

Exercícios:  
  
Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.

Interface gráfica do usuário, Texto, Aplicativo, Email

O conteúdo gerado por IA pode estar incorreto.