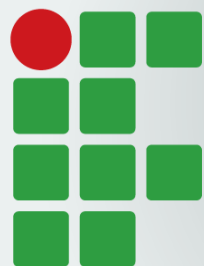


AULA 4 – Padrão de Projetos

Prof. Igor Justino Rodrigues
06/11/2024





**INSTITUTO
FEDERAL**

Goiano

Campus
Ceres

OBJETIVOS DA AULA

- Compreender o conceito e a motivação do padrão Builder.
- Identificar cenários onde o Builder é uma solução adequada.
- Implementar o padrão em código para criar objetos complexos de forma controlada.

17/10/2024

Padrão *Builder*

- O padrão *Builder* é um padrão de criação que permite construir objetos complexos passo a passo. Ao invés de usar um único construtor com muitos parâmetros, o *Builder* organiza a construção do objeto em etapas menores e mais fáceis de gerenciar.



Comparando

Diferença em Relação ao Singleton e Factory:

O **Singleton** controla a criação de uma única instância de uma classe (exemplo: uma conexão com banco de dados).

O **Factory** cria objetos de diferentes tipos sem expor a lógica de criação.

O **Builder**, por outro lado, é ideal para objetos complexos e personalizáveis, onde os atributos podem variar muito e devem ser construídos de maneira gradual.

O *Builder* é ideal para casos em que o objeto possui muitos atributos opcionais ou diferentes configurações.



Exemplo de Uso - Criação de um Perfil de Usuário

Imagine que queremos criar um perfil de usuário que pode ter várias informações, mas algumas são opcionais. Um construtor tradicional poderia deixar o código confuso:

```
$usuario = new Usuario("João", "joao@example.com", null,  
null, "12345-6789", null, "Rua A, 123", null);
```

Exemplo de Uso - Criação de um Perfil de Usuário

Com o Builder, a construção fica mais clara:

```
$usuarioBuilder = new UsuarioBuilder();  
$usuario = $usuarioBuilder->setNome("João")  
->setEmail("joao@example.com ")  
->setTelefone("12345-6789")  
->setEndereco("Rua A, 123")  
->getUsuario();
```



Características do Padrão Builder

Principais Características

Separação de Construção e Representação:

- O *Builder* divide a construção do objeto em vários métodos, facilitando a leitura e organização do código.


Principais Características

Encadeamento de Métodos (Method Chaining): Cada método retorna o *Builder*, permitindo o encadeamento de chamadas, como em:

```
$usuarioBuilder->setNome("João")->setEmail("joao@example.com")->getUsuario();
```


Principais Características

Flexibilidade na Criação:



Diferentes configurações de um objeto podem ser criadas com o mesmo *Builder*, ideal para quando queremos uma "receita" flexível.

HORA DE PARTICAR

