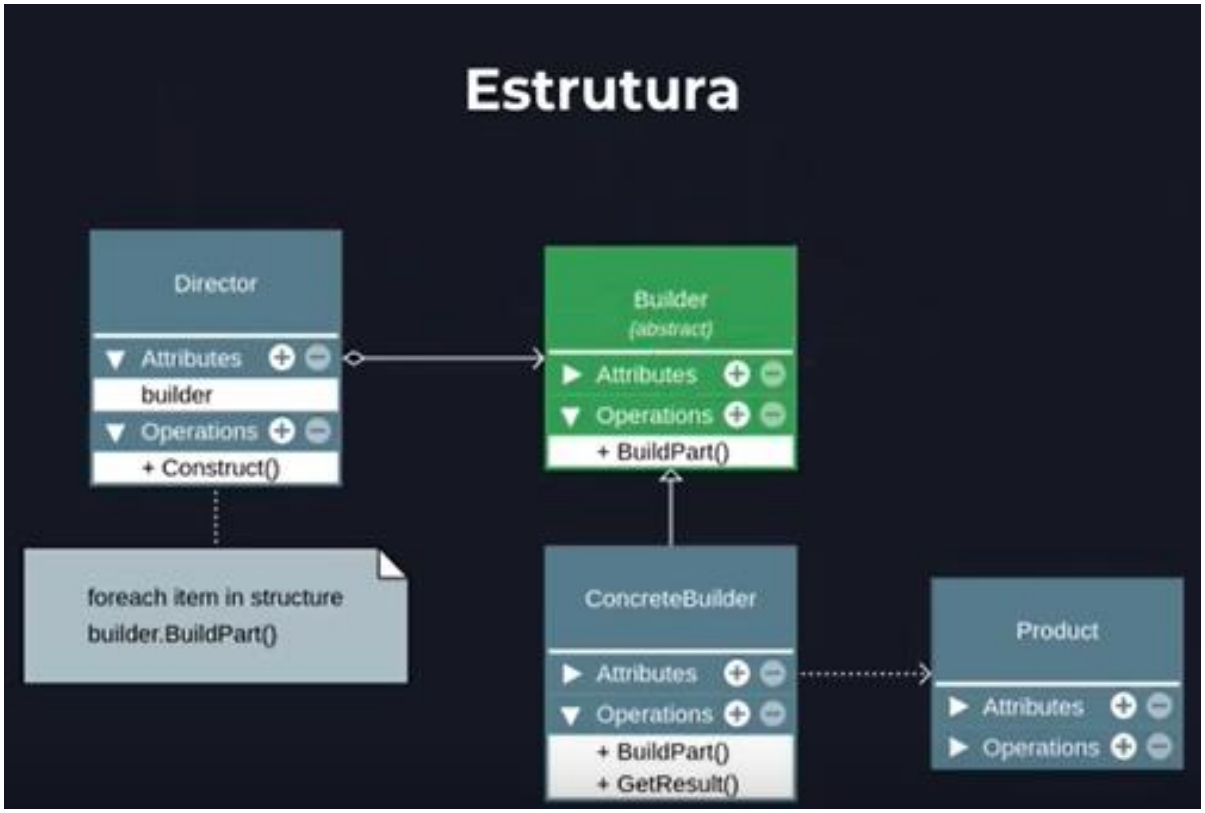
Builder


Estrutura
Director
Attributes
builder
Operations
+ Construct()
Builder
(abstract)
Attributes
Operations
+ BuildPart()
foreach item in structure
builder.BuildPart()
ConcreteBuilder
Attributes
Operations
+ BuildPart()
+ GetResult()
Product
Attributes
Operations

Facilita Criação de Objetos

Usado para facilitar a criação de objetos, separando a criação de outras funções gerais. Também podem ser definidos padrões e regras dentro do builder.

Director

Usado para organizar a criação de um objeto, algumas vezes é necessários uma ordem na criação de um objeto e essa é a função do director. Ordenar a criação do objeto atuando sobre o builder.

Builder

(interface ou abstract)

Usado como um contrato para um ou mais builders, quando houver mais de um builder pode ser interessante que todos tenham herança nessa interface base.

ConcreteBuilder

É a classe concreta que implementa as funções e regras do builder, nessa classe existem as regras, padrões e o objeto a ser construído.

Product

É a classe sobre a qual o builder atua na construção.

Vantagens

- Facilita criação de objetos
- Permite construção de objetos complexos passo a passo.
- Organiza a criação de objetos
- Não permite acesso do produto durante sua construção
- Permite Flexibilidade de construtor

Exemplo

Classe produto

```
class Exemplo {
private:
    int _a;
    string _b;
public:
    void setA(int a):_a(a){}
    void setB(string b): _b(b){}

};
```

Classe ConcreteBuilder

```
class ExemploBuider{
private:
    Exemplo _exemplo;
public:
    ExemploBuider(): _exemplo(Exemplo()){}
    ExemploBuider *setA(int a){
        _a = a;
        return this;
    }
    ExemploBuider *setB(string b){
        _b = b;
        return this;
    }
    Exemplo buid(){
        return _exemplo;
    }
};
```