Builder

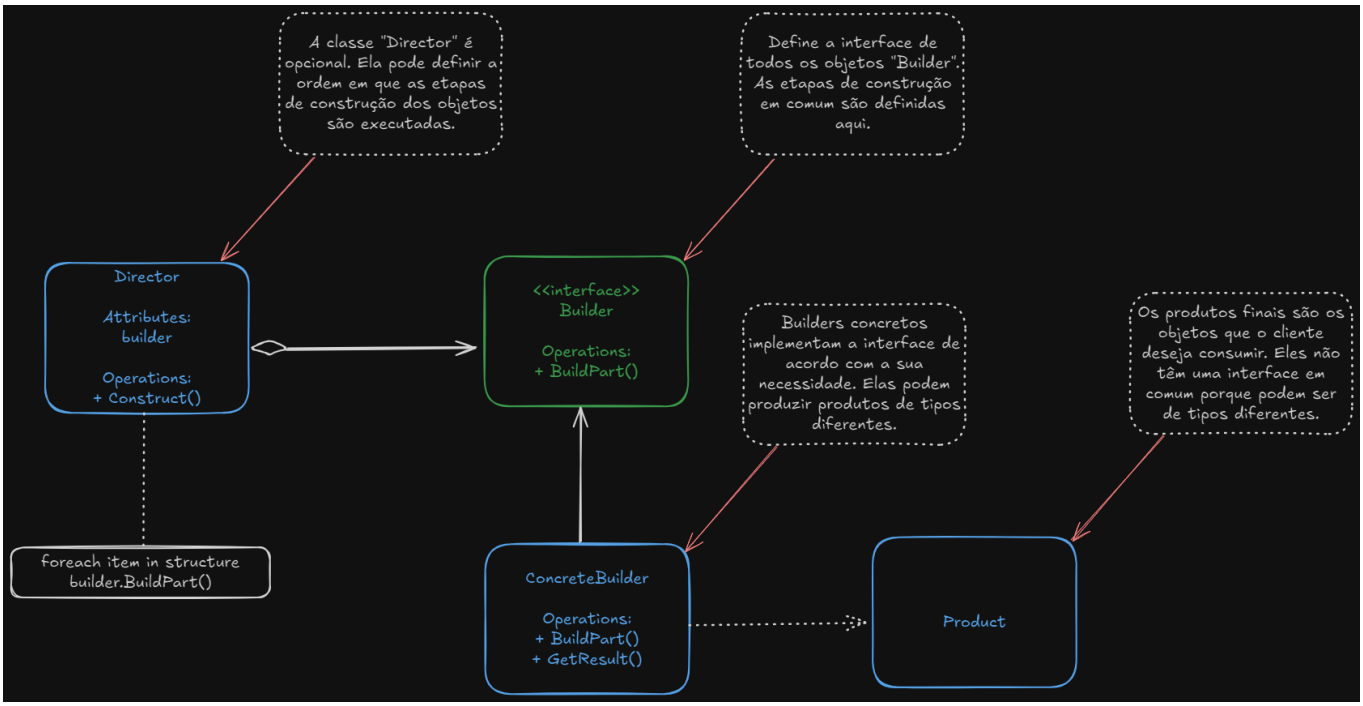
Intenção: Separar a construção de um objeto complexo da sua representação de modo que o mesmo processo de construção possa criar diferentes representações.

Tipo: creational

Visão Geral

- O padrão sugere a separação do código que cria e o código que usa o objeto.
- Trata da criação de objetos complexos (complexos de verdade)
  - Construtores muito complexos
  - Composição de vários objetos (composite)
  - Algoritmo de criação de objeto complexo
- Permite a criação de um objeto em etapas
- Permite *method chaining* (chamada de métodos uma após a outra)
- O objeto final pode variar de acordo com a necessidade
- É um padrão complexo

Estrutura



Implementação

ESTE EXEMPLO, NÃO É UM CASO ONDE VOCÊ UTILIZARIA O BUILDER

```
public class Person
{
    public string? Name;
    public int Age;
}

public class PersonBuilder
{
    private Person _person = new();

    public void NewPerson()
    {
        _person = new Person();
    }

    public void SetName(string name)
    {
        _person.Name = name;
    }

    public void SetAge(int age)
    {
        _person.Age = age;
    }

    public Person GetResult()
    {
        return _person;
    }
}

/*
var personBuilder = new PersonBuilder();
var person1 = personBuilder.NewPerson()
    .SetName("John")
    .SetAge(30)
    .GetResult();
```

```
personBuilder.NewPerson();

var person2 = personBuilder.SetName("Jane")
    .SetAge(25)
    .GetResult();
*/
```

Consequências

Bom:

- Separa a criação de utilização
- O cliente não precisa criar objetos diretamente
- O mesmo código pode construir objetos diferentes
- Ajuda na aplicação dos princípios SRP e OCP

Ruim:

- O código final pode se tornar muito complexo