SENAI Serviço Nacional de Aprendizagem Industrial

Dominando a POO em Dart

Bem-vindos à nossa jornada pela Programação Orientada a Objetos em Dart. Preparem-se para desvendar os pilares que sustentam o desenvolvimento de software moderno e eficiente.

Produzido por MARCIO SCHOENFELDER

Agenda da Nossa Jornada

1

Fundamentos da POO

Classes, objetos e seus componentes essenciais.

2

Encapsulamento em Ação

Protegendo dados com variáveis privadas e métodos de acesso.

3

Herança e Polimorfismo

Reutilização de código e flexibilidade.

4

Classes Abstratas e Interfaces

Definindo contratos e estruturas.

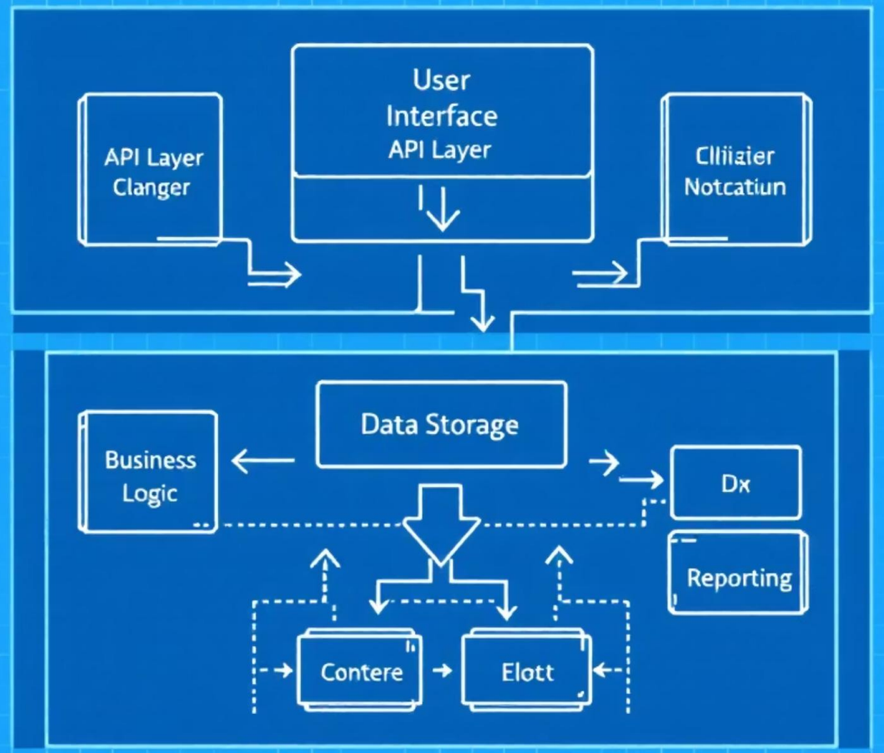
5

Tratamento de Exceções

Lidando com erros de forma elegante.

Introdução aos Fundamentos da POO

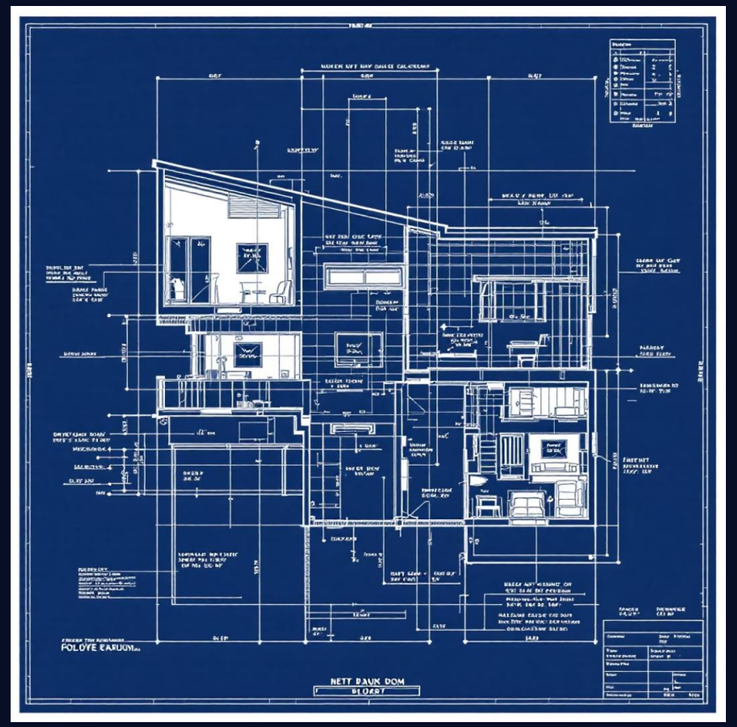
Nesta aula, desvendaremos os conceitos essenciais da Programação Orientada a Objetos em Dart, estabelecendo a base para a construção de softwares robustos e organizados.



O Coração da POO: Objeto, Classe e Instância

Classe

Um **molde** ou **planta** que descreve as características (atributos) e comportamentos (métodos) de um tipo de objeto.



Objeto

Uma **entidade** no mundo real ou no software, que é criada a partir de uma classe. É uma representação de algo.

Instância

Um **objeto específico** criado a partir de uma classe. Cada instância possui seus próprios valores para os atributos.

Anatomia de uma Classe: Atributos e Métodos

Atributos

As **características** ou **propriedades** que um objeto possui. Pense neles como as variáveis dentro de uma classe que armazenam o estado de um objeto. Ex: nome, idade, cor.

Métodos

As **ações** ou **comportamentos** que um objeto pode realizar. São funções que operam nos dados do objeto. Ex: andar(), comer(), calcularArea().

A combinação de atributos e métodos define completamente o que um objeto "é" e o que ele "faz".

Produzido por MARCIO SCHOENFELDER

Construindo Objetos: Construtores em Dart

Construtores são métodos especiais que permitem inicializar os atributos de um objeto quando ele é criado. Em Dart, existem vários tipos de construtores.

Exemplo Prático

```
class Pessoa {
    String nome;
    int idade;

    // Construtor padrão
       Pessoa(this.nome, this.idade);

    void apresentar() {
       print('Olá, meu nome é $nome e tenho $idade anos.’);
    }

    }
```

Encapsulamento e Controle de Acesso

O encapsulamento é um dos princípios fundamentais da POO, garantindo que o estado interno de um objeto seja protegido e acessado de forma controlada.

Protegendo Dados com Variáveis Privadas (_)

Em Dart, usamos o prefixo underscore (_) para indicar que uma variável é privada a uma biblioteca (arquivo). Isso ajuda a encapsular o estado do objeto.

Variáveis Privadas

```
class ContaBancaria {
   double _saldo; // Variável privada

ContaBancaria(this._saldo);

  void depositar(double valor) {
    _saldo += valor;
  }
}
```

Getters e Setters: Portões de Acesso aos Dados

Para acessar e modificar variáveis privadas de forma controlada, utilizamos Getters (para ler) e Setters (para escrever). Eles oferecem validação e lógica adicional.

Getters

Permitem ler o valor de uma propriedade privada. Semelhante a uma função que retorna um valor, mas sem parênteses.

```
double get saldo => _saldo;
```

Setters

Permitem modificar o valor de uma propriedade privada. Permite adicionar lógica de validação antes da atribuição.

```
set saldo(double novoSaldo) {
  if (novoSaldo >= 0) {
   _saldo = novoSaldo;
  }
}
```

Próximos Passos: Codificando e Explorando

- Crie uma classe `Carro` com atributos `marca`, `modelo` e `ano`.
- Crie um construtor e um método `exibirDetalhes()`.
- Instancie dois objetos e chame o método.
- Crie uma classe Produto com atributo privado _preco, use getter e setter para validar que o preço nunca seja negativo.

Continuem praticando!

A maestria em POO vem com a experiência.