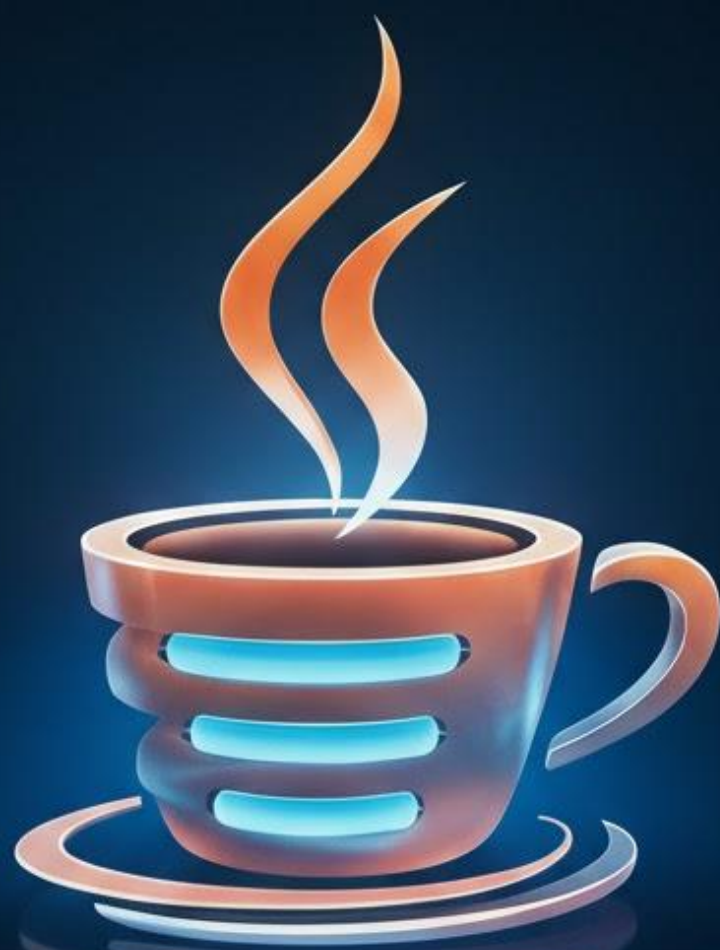
JAVA RELOADED

O PODER DA PROGRAMAÇÃO ORIENTADA A OBJETOS

JAVA



DOMINANDO A PLATFORMA

JULIANA BENEDETTI

Introdução

O Renascimento do Java

Java está mais vivo do que nunca. Mesmo depois de décadas, ele continua sendo uma das linguagens mais utilizadas no mundo — presente em sistemas bancários, aplicativos Android, plataformas web e até dispositivos inteligentes.

Mas o segredo do sucesso do Java não é apenas sua estabilidade.É a Programação Orientada a Objetos (POO) — um conceito poderoso que permite criar códigos organizados, reutilizáveis e parecidos com o mundo real.Neste eBook, você vai entender como o Java funciona de forma simples e prática.

Vamos percorrer desde o básico até os principais conceitos da POO, sempre com exemplos curtos e reais que você pode testar agora mesmo.

Prepare-se para recarregar seu conhecimento em Java

🚀Capítulo 1

O Começo da Jornada: Entendendo o Java

O que é e por que é tão usado?Java é uma linguagem multiplataforma — ou seja, você escreve uma vez e executa em qualquer sistema.Ela é segura, robusta e ideal para quem quer aprender a programar com base sólida.

Exemplo prático: o clássico “Olá Mundo”

Untitled-1

```java
public class OlaMundo {
    public static void main(String[] args) {
        System.out.println("Olá, mundo do Java!");
    }
}
```

Tudo em Java acontece dentro de uma classe e de um método principal (main), onde o programa começa a rodar.

⚙ Capítulo 2

Pensando em Objetos: A Base da POO

A Programação Orientada a Objetos (POO) é a alma do Java.Ela nos permite criar modelos (classes) que representam coisas do mundo real — como pessoas, carros ou produtos — e depois gerar objetos a partir desses modelos.

Exemplo real: um sistema de cadastro de alunos

Untitled-1

```java
public class Aluno {
    String nome;
    int idade;

    void exibirInfo() {
        System.out.println("Nome: " + nome + ", Idade: " + idade);
    }
}

// Criando e usando o objeto
public class Main {
    public static void main(String[] args) {
        Aluno aluno1 = new Aluno();
        aluno1.nome = "Ana";
        aluno1.idade = 20;
        aluno1.exibirInfo();
    }
}
```

Aqui, “Aluno” é o molde (classe), e “aluno1” é o objeto que usamos para armazenar dados reais.

🎲 Capítulo 3

Encapsulamento: Protegendo seus Dados

Encapsular significa proteger as informações internas de uma classe, permitindo o acesso apenas por meio de métodos controlados.É como colocar uma senha para que ninguém mexa nos dados diretamente.

Exemplo real: controle de conta bancária

Untitled-1

```
public class ContaBancaria {
    private double saldo;

    public void depositar(double valor) {
        saldo += valor;
    }

    public double getSaldo() {
        return saldo;
    }
}

public class Main {
    public static void main(String[] args) {
        ContaBancaria conta = new ContaBancaria();
        conta.depositar(500);
        System.out.println("Saldo atual: R$" + conta.getSaldo());
    }
}
```

Assim, o saldo é alterado apenas pelos métodos definidos, evitando erros e fraudes.

🎯 Capítulo 4

Herança e Polimorfismo: Reaproveitando e Adaptando Código

Herança permite que uma classe “filha” herde características de uma “classe mãe”.Polimorfismo significa “muitas formas” um mesmo método pode se comportar de maneiras diferentes.

Exemplo real: cadastro de funcionários

Untitled-1

```java
class Funcionario {
    void trabalhar() {
        System.out.println("Trabalhando...");
    }
}

class Gerente extends Funcionario {
    void trabalhar() {
        System.out.println("Gerenciando a equipe...");
    }
}

public class Main {
    public static void main(String[] args) {
        Funcionario f1 = new Funcionario();
        Funcionario f2 = new Gerente();

        f1.trabalhar();
        f2.trabalhar(); // Polimorfismo em ação!
    }
}
```

O método “trabalhar” é o mesmo, mas o comportamento muda conforme o tipo do objeto.

Capítulo 5

Mãos à Obra: Projeto Final com POO

Agora vamos aplicar tudo que aprendemos em um mini projeto prático.Nosso desafio: criar um sistema simples de cadastro de produtos.

Exemplo prático:

Untitled-1

```java
class Produto {
    private String nome;
    private double preco;

    public Produto(String nome, double preco) {
        this.nome = nome;
        this.preco = preco;
    }

    public void exibirInfo() {
        System.out.println("Produto: " + nome + " | Preço: R$" + preco);
    }
}

public class Main {
    public static void main(String[] args) {
        Produto p1 = new Produto("Notebook", 3500.00);
        Produto p2 = new Produto("Mouse", 80.00);

        p1.exibirInfo();
        p2.exibirInfo();
    }
}
```

Aqui temos classe, construtor, encapsulamento e criação de objetos reais.Um pequeno passo, mas já com a base para sistemas maiores.

🏁 Conclusão

O Java Está Só Começando

Você acabou de dar um passo importante no universo da programação.Aprendeu o essencial sobre Java, entendeu a base da Programação Orientada a Objetos e viu exemplos aplicáveis ao dia a dia.

Mas isso é só o começo.
Com esses fundamentos, você pode evoluir para criar interfaces gráficas, APIs, aplicativos Android e muito mais.

O poder do Java está em suas mãos.
Agora, é hora de praticar, testar e continuar recarregando seu conhecimento.