Programação Avançada – AULA 01

Matheus Moresco

Análise e Desenvolvimento de Sistemas - 5º Período

2025/01

Informações da Disciplina

- **Disciplina:** Programação Avançada
- **Curso:** Engenharia de Software/Análise e Desenvolvimento de Sistemas
- **Carga Horária:** 80 horas
- **Aulas por Semana:** 4 (2 encontros de 1 hora e 40 minutos cada)
- **Professor:** Matheus Moresco
- **Semestre:** 2025/01

Introdução a disciplina

- A disciplina de **Programação Avançada** aprofunda os conceitos essenciais para o desenvolvimento de sistemas robustos, escaláveis e eficientes.
- Técnicas avançadas da **Programação Orientada a Objetos (POO)**, como sobrecarga, classes abstratas e interfaces, além de aprender a manipular **coleções de dados**, construir **interfaces gráficas**, lidar com **tratamento de exceções** e implementar **persistência de dados** com JDBC e JPA.
- Base para o desenvolvimento de **sistemas modulares, reutilizáveis e bem estruturados**.

Materiais

- **Linguagem de programação**: Java
- **IDE**: Eclipse, IntelliJ IDEA, VS Code, NetBeans
- **Banco de Dados**: MySQL ou PostgreSQL
- **Frameworks**: Java Swing, JDBC, JPA

- **Bibliografia**:
  - HORSTMANN, Cay S.; FURMANKIEWICZ, Edson. Big Java / 2006 Porto Alegre: Bookman, 2006.
  - SCHILDT, Herbert; SILVA, Aldir Coelho Corrêa da. Java para iniciantes - 5. ed. / 2013 Porto Alegre: Bookman, 2013.
  - DEITEL, Harvey M; DEITEL, Paul J; DEITEL, Abbey. Android : como programar - 2 / 2015 Porto Alegre: Bookman, 2015.

Programação

1. **Revisão e Fundamentos Avançados da POO**
   - Revisão de POO
   - Sobrecarga de Métodos e Construtores
   - Classes Abstratas e Interfaces
   - Modificadores de Acesso e Princípios de Encapsulamento

2. **Estruturas de Dados e Manipulação de Coleções**
   - Arrays e Listas em Java
   - Conjuntos e Mapas
   - Streams e Expressões Lambda

Programação

3. Tratamento de Exceções e Boas Práticas

- Tipos de Exceções
- Uso do Try-Catch-Finally
- Exceções Personalizadas e Lançamento de Erros

4. Persistência de Dados e Banco de Dados

- Conexão com Banco de Dados usando JDBC
- Manipulação de Dados com JPA
- ORM e Mapeamento Objeto-Relacional

Programação

5. Desenvolvimento de Interfaces Gráficas com Swing

- Componentes Gráficos e Layouts
- Tratamento de Eventos e Interação com Usuário
- Integração da Interface com Banco de Dados

6. Projeto Final e Aplicação Prática

- Planejamento e Implementação do Mini-Projeto
- Refinamento, Testes e Revisão
- Apresentação e Avaliação Final

Metodologia

- Aulas teóricas expositivas
- Exercícios práticos e mini-projetos
- Aulas práticas no Laboratórios de programação
- Projeto Final

Avaliação

- Prova Pratica/ Projeto Final: 80%
- Atividade de Estudo Programada(AEP): 10%
- Prova integrada: 10%

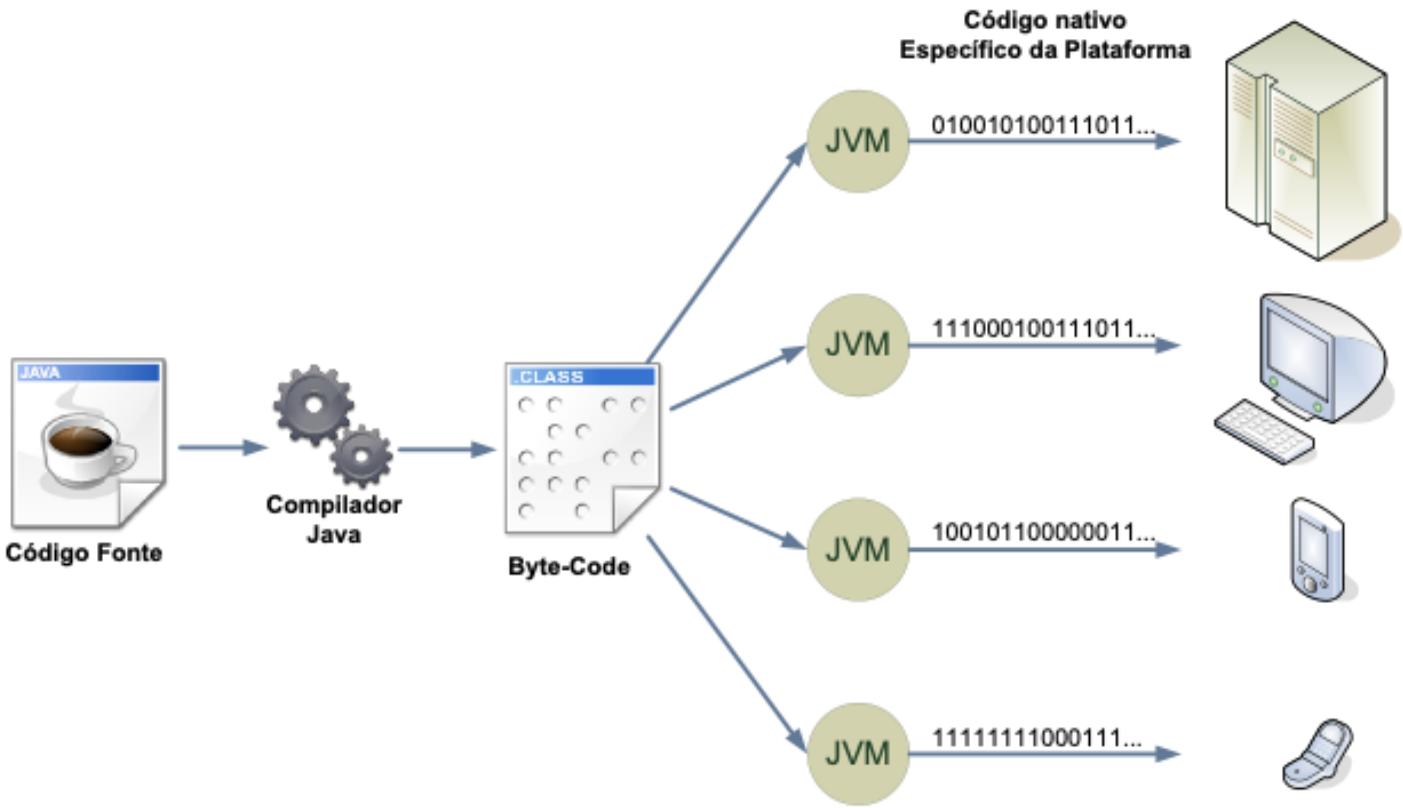
O que é Java?

- Java é uma linguagem de programação orientada a objetos e multiplataforma, criada pela Sun Microsystems (agora parte da Oracle) em 1995.
- Ela é conhecida pelo lema "**Write Once, Run Anywhere**" (escreva uma vez, execute em qualquer lugar), pois o código Java é compilado para **bytecode**, que pode ser executado em qualquer dispositivo que tenha a **Java Virtual Machine (JVM)**.

Como Java Funciona?

- **Compilação** – O código-fonte é escrito em **arquivos.java** e compilado para **bytecode (.class)** usando o compilador **javac**.
- **Execução na JVM** – O bytecode é interpretado e executado pela **Java Virtual Machine (JVM)**, permitindo compatibilidade com diversos sistemas operacionais.
- **Bibliotecas e APIs** – Java possui um vasto conjunto de bibliotecas padrão (Java API) que facilitam o desenvolvimento de aplicações.

Como Java Funciona?



Vantagens do Java

- **Portabilidade** – O mesmo código pode rodar em diferentes sistemas operacionais sem modificações.
- **Orientação a Objetos** – Facilita o design modular e reutilização de código.
- **Fortemente Tipada** – Necessita da declaração de tipo de todas as estruturas.
- **Segurança** – A JVM oferece proteção contra execução de código malicioso.
- **Gerenciamento de Memória Automático** – O **Garbage Collector** gerencia a alocação e liberação de memória.
- **Multithreading** – Suporte nativo para execução de múltiplas tarefas simultaneamente.
- **Ampla Adoção no Mercado** – Usado em desenvolvimento Web, Mobile (Android), sistemas empresariais e aplicações de alto desempenho.

JDK vs. JRE: Qual a diferença?

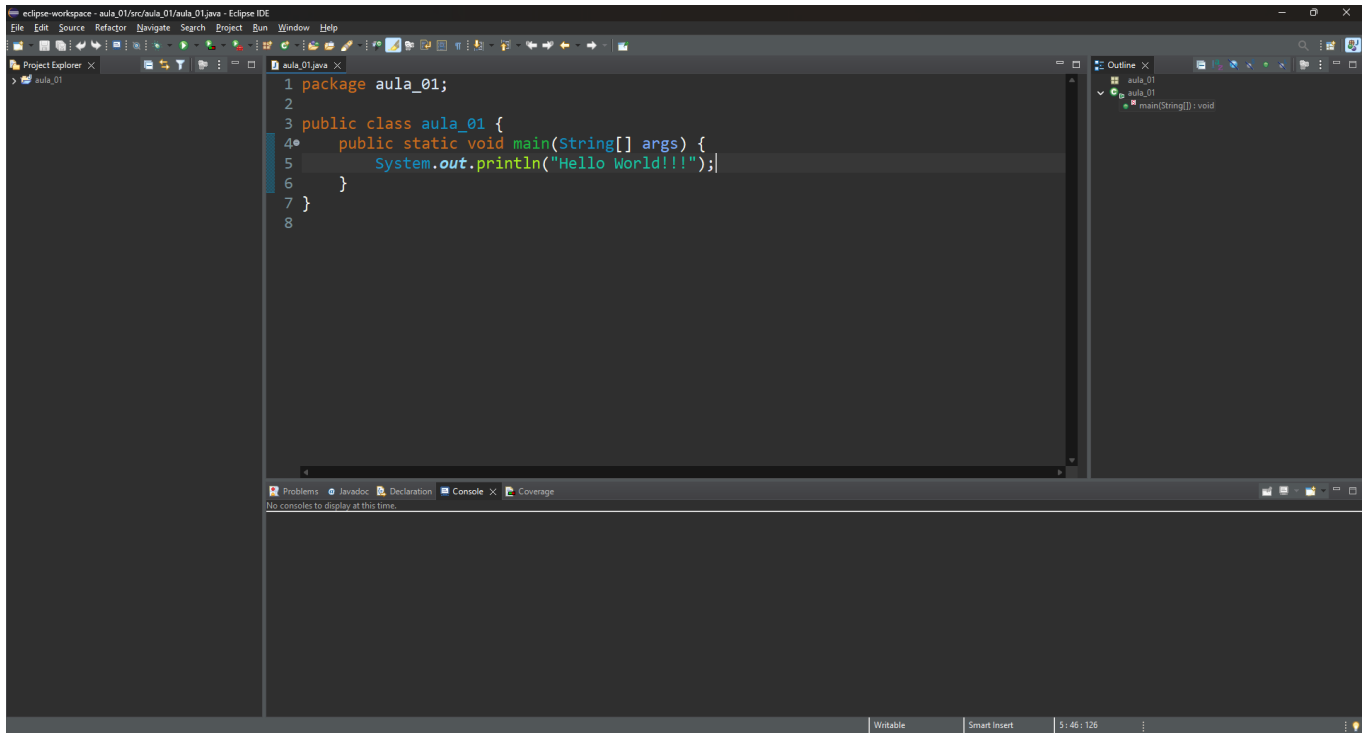
- **JDK (Java Development Kit)**
  - Pacote completo para desenvolvimento de aplicações Java.
  - Inclui o JRE, além de compilador (javac), depurador (jdb) e ferramentas de desenvolvimento.
  - Necessário para criação e compilação de programas em Java.

- **JRE (Java Runtime Environment)**
  - Ambiente necessário apenas para executar aplicações Java.
  - Contém a JVM (Java Virtual Machine), bibliotecas e arquivos de suporte.
  - Não permite compilar código, apenas rodar programas Java já compilados.

Instalação

- **O JDK, abreviação para Java Development Kit, é um conjunto de utilitários cuja a finalidade é a permissão para criação de jogos e programas para a plataforma Java**.
- **O Java JDK é composto pelo**
  - **compilador**
  - **bibliotecas (API's)**
  - Máquina Virtual Java
- O JDK dispõe de um arquivo executável que faz todo o trabalho de instalação e configuração do ambiente.
- Download

Introdução ao Eclipse



Download

Exercícios

1. Instale o JDK no seu computador.
2. Escolha e instale uma IDE para trabalhar com Java.
3. Crie um projeto, crie a classe main e faça o programa printar uma mensagem no console