

# Simulador de Sistema de Arquivos com Journaling

---

**Disciplina:** Proj. de Sistema Operacional

**Alunos:** Ana Beatriz de Sousa Colaço e Matheus Holanda de Sá

## Metodologia

O simulador foi desenvolvido em linguagem de programação Java. Ele funciona em "Modo Avançado" (Shell), onde o sistema roda em um loop contínuo aguardando comandos. O programa recebe as chamadas de métodos com os devidos parâmetros, executa a funcionalidade correspondente a um comando de SO (como `mkdir` ou `copy`) e exibe o resultado na tela, mantendo a persistência dos dados em disco.

---

## Parte 1: Introdução ao Sistema de Arquivos com Journaling

### Descrição do sistema de arquivos

Um sistema de arquivos é o conjunto de estruturas lógicas que permite ao sistema operacional controlar o acesso e armazenamento de dados no disco. Ele define como os arquivos são nomeados, armazenados e organizados. Neste projeto, simulamos essa organização através de uma hierarquia lógica (árvore de diretórios).

### Journaling

O Journaling é uma técnica utilizada para manter a consistência do sistema de arquivos em caso de falhas.

- **Propósito:** Registrar as operações em um log antes (ou durante) a sua execução real.
  - **Funcionamento:** Implementamos um modelo onde cada operação de modificação (criação, exclusão, renomeação) é registrada sequencialmente em um arquivo de texto (`journal.txt`). Além disso, o estado atual do sistema (snapshot) é persistido em um arquivo binário (`filesystem.dat`) após cada comando, garantindo que os dados não sejam perdidos ao fechar o simulador.
- 

## Parte 2: Arquitetura do Simulador

### Estrutura de Dados

Utilizamos classes Java para representar os elementos do sistema, organizados em uma estrutura de árvore:

- **FileSystemSimulator (Kernel):** Gerencia a memória e o fluxo de comandos.
- **Directory (Nó):** Contém uma lista de subdiretórios (`List<Directory>`) e uma lista de arquivos (`List<File>`).
- **File (Folha):** Representa o arquivo final com nome e conteúdo.
- **Persistência:** Utilizamos a interface `java.io.Serializable` para converter toda a estrutura de objetos em bytes e salvá-la no arquivo `filesystem.dat`.

### Implementação do Journaling

O Journaling foi implementado através da classe `Journal`.

- **Estrutura do Log:** Arquivo de texto (`journal.txt`) onde cada linha contém um Timestamp e o Comando executado. Exemplo: `[2025-12-05 20:30:00] CMD: MKDIR docs`.
  - **Operações Registradas:** Todas as operações que alteram a estrutura do disco (`MKDIR`, `TOUCH`, `RMDIR`, `DEL`, `REN`, `COPY`) são gravadas no log. Operações de apenas leitura (`DIR`, `CD`) são executadas em memória sem gerar entrada no log de transações.
- 

## Parte 3: Implementação em Java

### Classe "FileSystemSimulator"

É a classe principal que implementa o simulador. Contém o método `main` (loop do shell) e os métodos lógicos para cada operação (`createDirectory`, `deleteFile`, `copy`, `rename`, etc.). Ela é responsável por carregar os dados ao iniciar e salvar ao modificar.

### Classes File e Directory

- **File:** Classe simples que armazena o nome do arquivo (`filename`) e seu conteúdo (`content`).
- **Directory:** Classe complexa que gerencia as listas de filhos e o ponteiro para o diretório pai (`parent`), permitindo a navegação `..` (voltar).

### Classe Journal

Gerencia o log de operações. Utiliza `FileWriter` em modo *append* para garantir que o histórico de comandos seja preservado entre diferentes execuções do programa.

---

## Parte 4: Instalação e funcionamento

### Passo a Passo para Execução

#### 1. Pré-requisitos:

- Ter o Java (JDK) instalado na máquina.
- Um editor de código ou terminal.

#### 2. Compilação:

No terminal, navegue até a pasta do projeto e execute:

```
javac *.java
```

#### 3. Execução:

Inicie o simulador com o comando:

```
java FileSystemSimulator
```

#### 4. Utilização (Comandos):

O sistema abrirá um shell simulado (`/>`). Utilize os comandos abaixo:

- `MKDIR <nome>`: Cria um novo diretório.

- **TOUCH <nome>**: Cria um arquivo vazio.
- **CD <nome>**: Entra em uma pasta (use **CD ..** para voltar).
- **DIR**: Lista o conteúdo da pasta atual.
- **DEL <nome>**: Apaga um arquivo.
- **RMDIR <nome>**: Apaga um diretório.
- **REN <antigo> <novo>**: Renomeia um arquivo ou pasta.
- **COPY <origem> <destino>**: Copia um arquivo. Se o destino for uma pasta, copia para dentro dela; se for um nome, cria um duplicata renomeada.
- **JOURNAL**: Exibe o histórico de logs salvo em disco.
- **EXIT**: Salva e sai do sistema.

Link do GitHub

**<https://github.com/matheus-html/OS-Filesystem-Simulator-Java>**