Relatório: Uso de ReadWriteLock para Controle de Acesso a Preços

1. Objetivo

Demonstrar o uso do `ReadWriteLock` em Java para controlar o acesso concorrente a informações de preços. O exemplo ilustra como múltiplas threads podem ler os preços simultaneamente, enquanto uma única thread modifica os valores de forma exclusiva, garantindo a integridade dos dados.

2. Estrutura do Código

- **Escritor**:
  Responsável por modificar os preços. Executa três atualizações, utilizando bloqueio de escrita para garantir que as alterações sejam feitas sem interferência.
- **InfoPrecos**:
  Armazena os preços (preco1 e preco2) e utiliza `ReentrantReadWriteLock` para sincronizar as operações de leitura e escrita. Os métodos de leitura (`getPreco1` e `getPreco2`) adquirem o bloqueio de leitura, permitindo acesso concorrente, enquanto o método de escrita (`setPrices`) utiliza o bloqueio de escrita, impedindo acesso simultâneo durante a atualização.
- **Leitor**:
  Responsável por ler os preços. Executa 10 leituras em cada thread, utilizando os métodos de leitura sincronizados da classe `InfoPrecos`.
- **Principal**:
  Inicializa o sistema criando 5 threads de leitores e 1 thread de escritor, permitindo a execução concorrente das operações de leitura e escrita.

3. Fluxo de Execução

1. A classe `Principal` cria uma instância de `InfoPrecos` com preços iniciais.
2. São criadas 5 threads para leitores, cada uma realizando 10 leituras dos preços.
3. Uma thread de escritor é criada e executa três atualizações nos preços, utilizando bloqueio de escrita para garantir exclusividade.
4. Durante a execução, o `ReadWriteLock` permite que várias threads realizem leituras simultâneas, mas bloqueia as operações de leitura enquanto o escritor atualiza os preços.
5. Assim, o sistema garante a integridade dos dados e evita condições de corrida.

4. Exemplo de Execução

Saída do Console: `text Writer: Attempt to modify the prices. at Sat Feb 10 15:30:12 BRT 2025 Writer: Prices have been modified.`

```
Preço 1 =======> Preço 2 <====== Thread-1: Price 1: 1.234567
Thread-1: Price 2: 2.345678 ...
```

5. Conclusão

Este exemplo demonstra a eficácia do ReadWriteLock na coordenação de operações concorrentes em um ambiente multithread. Ao permitir que várias threads realizem leituras simultâneas e garantindo exclusividade durante a escrita, o sistema mantém a integridade dos dados e evita conflitos de acesso. Essa abordagem é essencial para aplicações que exigem alta concorrência e consistência nos dados compartilhados.