

# AULA 4 - MULTITHREADING

GS1020 – Programação Orientada a Objetos II

Prof. Dr. Murillo G. Carneiro  
*[mgarneiro@ufu.br](mailto:mgarneiro@ufu.br)*



*Material baseado nos slides cedidos pelo Prof. Rafael D. Araújo (FACOM/UFU)*

# O cenário mais simples

- Concentrar nossa atenção na realização de uma única tarefa de cada vez e fazer isso muitíssimo bem
  - Como seria cursar apenas uma disciplina por semestre?

# Concorrência e paralelismo

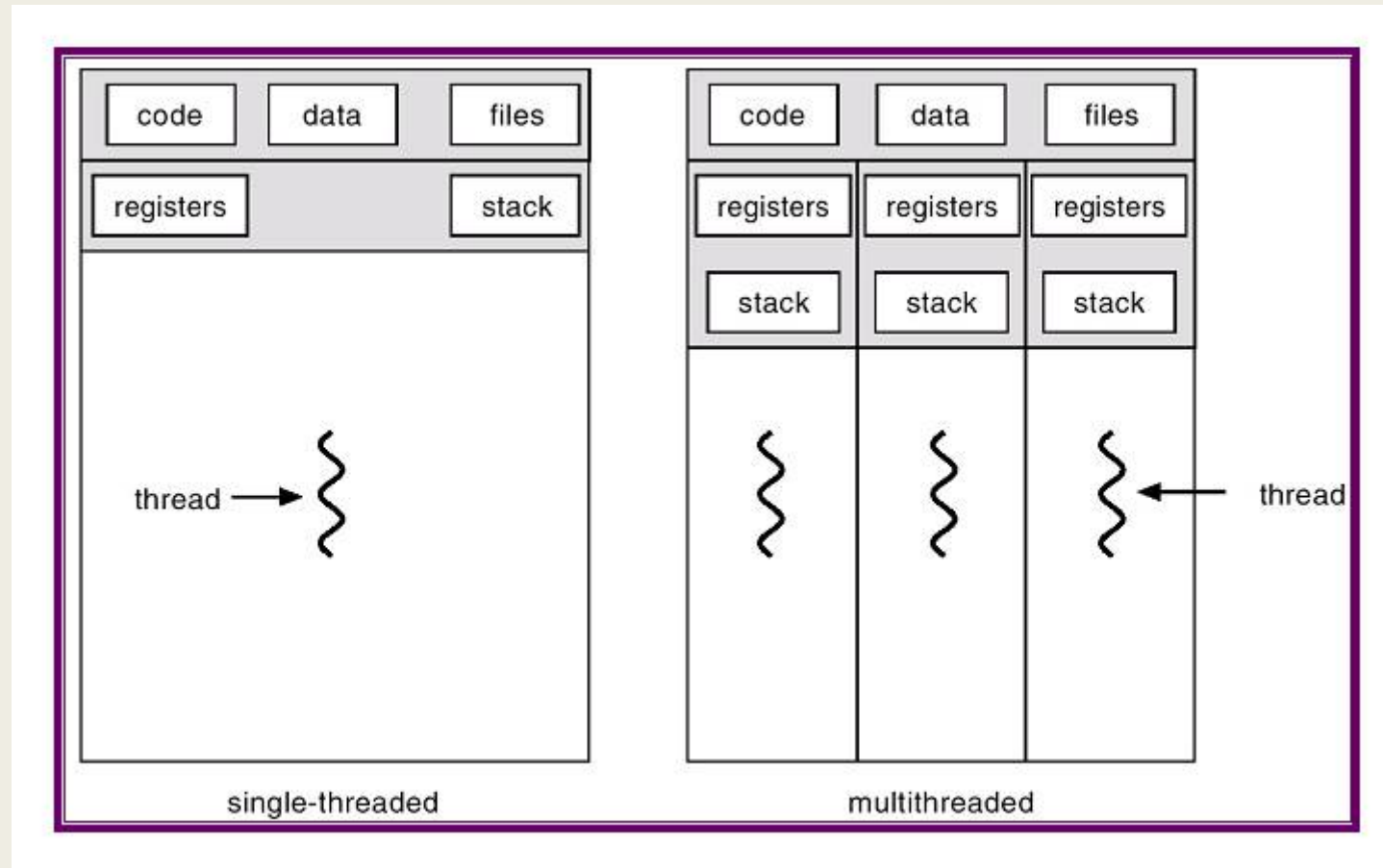
- Concorrência: habilidade para lidar com várias tarefas de uma vez.
  - Enquanto realiza um download, você ouve música, recebe emails e codifica a atividade de POO, tudo isso no mesmo computador.
- Paralelismo: habilidade para executar várias tarefas simultaneamente.
  - Cenário da concorrência no qual dispomos de arquiteturas computacionais multiprocessadas

# O que é uma *thread*?

- Um pequeno programa criado por um processo que pode ser executado concorrentemente com outras partes do mesmo processo
- As threads são independentes umas das outras e a execução de uma não afeta a execução de outra
  - *Podem trocar dados e informações entre si*
  - *Podem compartilhar os mesmos recursos do sistema (memória, por exemplo)*

progridem  
ao mesmo  
tempo

# Multithreading



Fonte: <http://recologia.com.br>

# Para que utilizar?

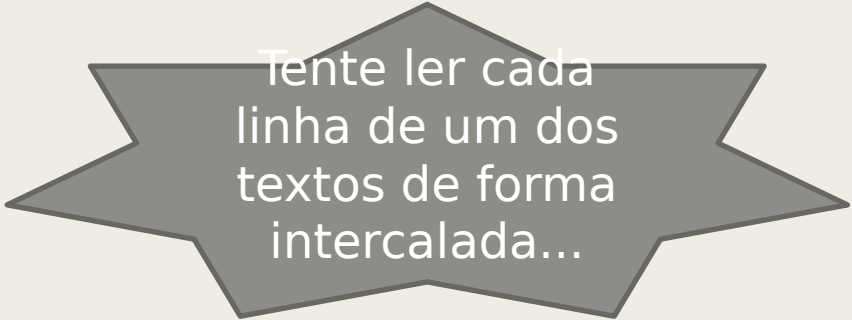
- Executar simultaneamente duas ou mais partes de um processo (programa) para maximizar a utilização do tempo de CPU

Ou seja, minimizar o tempo ocioso do processador

- Exemplos:

- *enquanto uma thread espera por uma operação de E/S, outra thread pode ser executada*
- *streaming de vídeo e áudio*
- *melhorar o desempenho da aplicação executando tarefas em background*

# Calma aí!



Tente ler cada  
linha de um dos  
textos de forma  
intercalada...

■ Escrever programas de múltiplas threads pode ser difícil!

De tudo, ficaram três coisas: a certeza de que ele estava sempre começando, a certeza de que era preciso continuar e a certeza de que seria interrompido antes de terminar. Fazer da interrupção um caminho novo. Fazer da queda um passo de dança, do medo uma escada, do sono uma ponte, da procura um encontro.

***O Encontro Marcado  
(Fernando Sabino)***

Confiança não se compra, não se desperta de uma hora para outra, não se pede. Confiança se constrói a passos lentos, na direção certa, no mesmo sentido, acrescentando valor às percepções alheias.

***Personal Branding:  
Construindo Sua Marca  
Pessoal  
(Arthur Bender)***

Há momentos em que as tribulações acontecem em nossas vidas, e não podemos evitá-las. Mas estão ali por algum motivo... É uma pergunta que não podemos responder ante – ou durante – as dificuldades. Só quando as ultrapassamos, entendemos por que estavam ali.

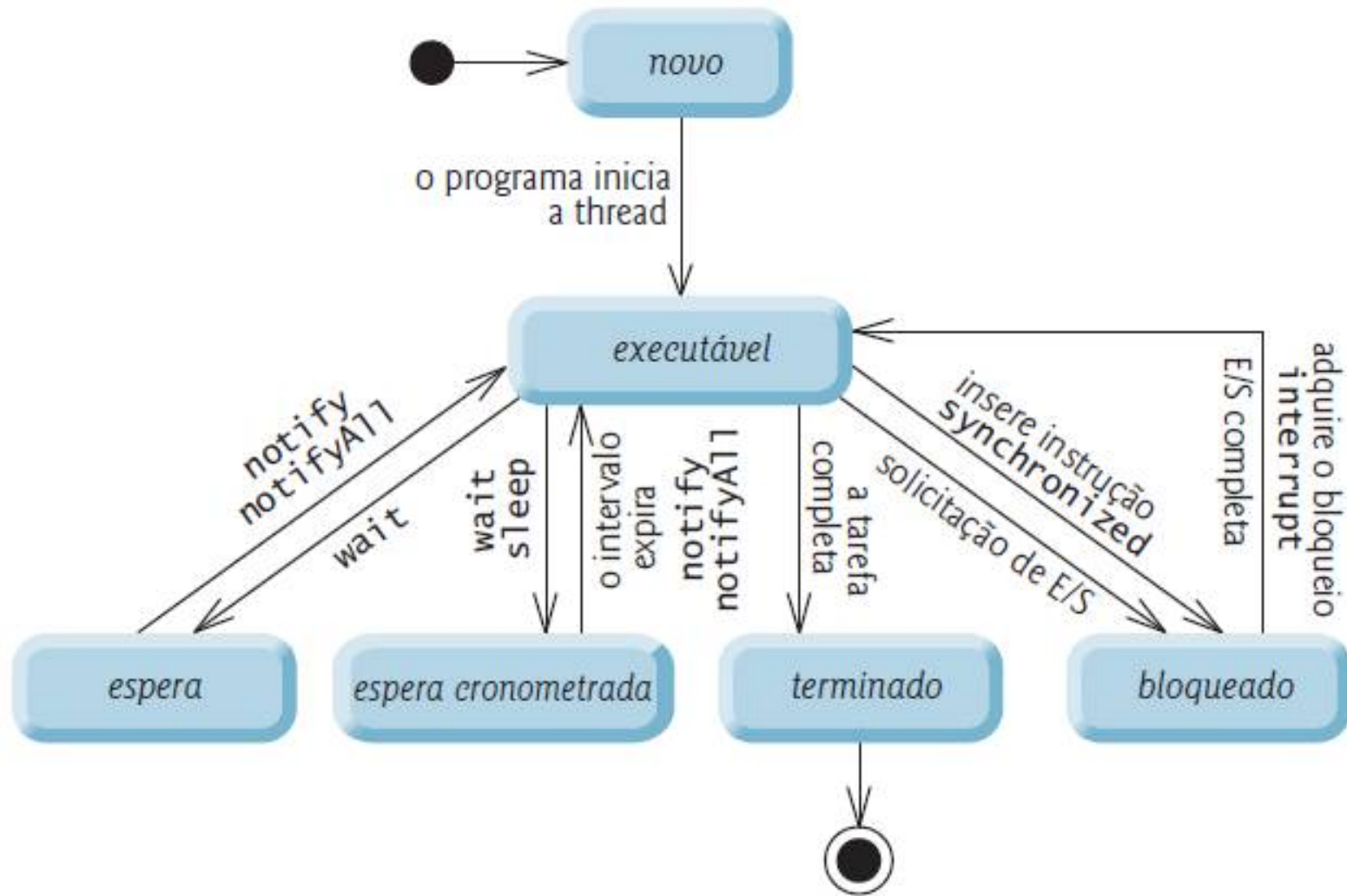
***O Monte Cinco  
(Paulo Coelho)***

# Implementação em Java

- Mecanismos de gerenciamento e sincronização de threads foram incorporados diretamente à linguagem (`java.lang`)
- Uma thread Java é representada por um objeto da classe `Thread` (`java.lang.Thread`)
- Ao executar uma aplicação Java, uma thread é criada e iniciada automaticamente para executar o método `main()`



# Ciclo de vida



# Implementação em Java

■ É possível criar uma thread explicitamente de duas maneiras:

- 1. Estendendo a classe Thread e instanciando um objeto desta nova classe*
- 2. Implementando a interface Runnable, passando um objeto desta nova classe como argumento para o construtor da classe Thread*

■ Nos dois casos, a thread é executada pelo método `run()`

# Alguns métodos da classe Thread

- `run()`: executa as atividades da thread. Quando esse método finaliza, a thread também finaliza.
- `start()`: dispara a execução de uma nova thread. Geralmente, antes de terminar, esse método chama o `run()`.
- `sleep(int x)`: coloca a thread no estado de espera por um tempo determinado (em milissegundos).
- `interrupt()`: interrompe a execução da thread.
- `interrupted()`: verifica se a thread está interrompida (estado bloqueado).
- `isAlive()`: verifica se a thread está viva (foi iniciada e ainda não morreu).

# Atividade de implementação

■ Implementar o exemplo contido em:

[https://www.tutorialspoint.com/java/java\\_multithreading.htm](https://www.tutorialspoint.com/java/java_multithreading.htm)

## 1. Usando Runnable

*1.1. Crie uma classe RunnableDemo que implementa a interface Runnable. Copie o código do exemplo.*

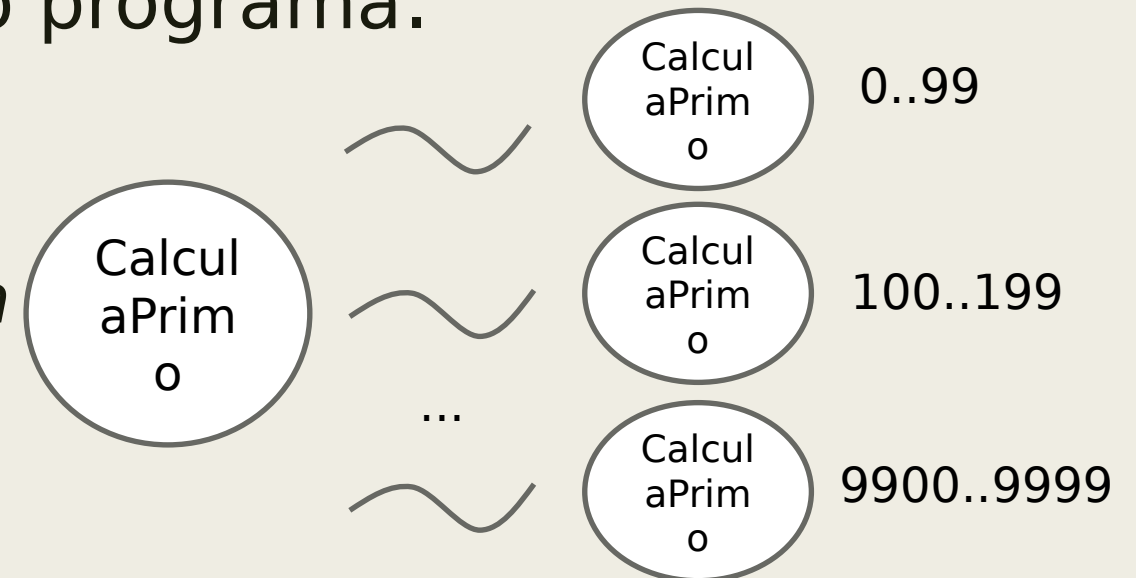
*1.2. No main, instancie a classe RunnableDemo.*

## Exercício (Entregar via Teams)

■ Implemente um programa que imprima os números primos existentes entre 0 e 9999. Calcule o tempo de execução do programa.

■ Altere o programa para utilizar Threads e calcule o tempo de execução do novo programa.

– *Para cada faixa de cem números, crie uma thread e dispare o processo para cada uma delas.*



# Referências

- DEITEL, H. M. Java: como programar, 8. ed. São Paulo: Prentice Hall, 2010. Capítulo 23.